ESD-TR-90-187

2

Technical Report
912

A Front End Filter Subsystem for an Adaptive Radar Signal Processor

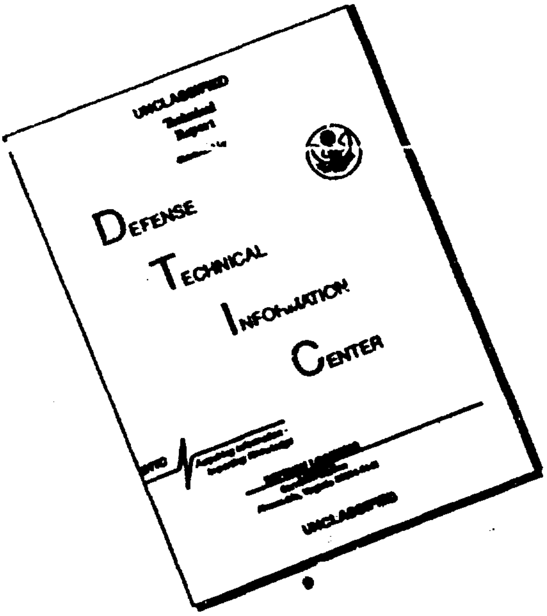
C.B. Robins

12 July 1991

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

Prepared for the Department of the Air Force
under Contract F19628-90-C-0002.

DISCLAIMER NOTICE



UNCLASSIFIED

Technical Report

DEFENSE

TECHNICAL

INFORMATION

CENTER

UNCLASSIFIED

THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Strategic Air Command, Department of the Air Force under Contract F19628-90-C-0002.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Hugh L. Southall

Hugh L. Southall, Lt. Col., USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

A FRONT END FILTER SUBSYSTEM FOR AN ADAPTIVE RADAR SIGNAL PROCESSOR

C.B. ROBINS
Group 27

TECHNICAL REPORT 912

12 JULY 1991

LEXINGTON MASSACHUSETTS

ABSTRACT

This report documents the front end subsystem portion of an eight-channel, adaptive nulling, radar signal processor test bed. The subsystem implements in excess of 12 billion operations per second on incoming data to effect signal conditioning through time-domain filtering. The hardware has been prototyped on eight circuit boards, each about 120 square inches, which are roughly half-populated. Compact packaging schemes are discussed in one of the appendices. This effort represents a demonstration of the technology required for a variety of on-board signal processors. Wherever possible, fault-tolerant design techniques and radiation-tolerant components have been used.

The front end subsystem receives eight channels of sampled data from the eight radar receiver A/D modules at the conversion rate of 4.5 MHz. The front end employs finite impulse response (FIR) filters to perform inphase and quadrature signal separation, channel equalization, and pulse compression. The coefficients for these filters are programmable via a VMEbus compatible interface. The front end also includes four digital beamformers; each computes a weighted sum of the eight channels of data on a sample-by-sample basis. The weights, also downloaded via the VMEbus interface, are computed and applied to perform adaptive nulling. The four resulting data streams, each with a rate of 0.75 million complex samples per second, are then output to the vector processor portion of the test bed for Doppler processing.

ACKNOWLEDGMENTS

I would like to express my gratitude to Joseph V. MacPhee, Steven C. Pohlig, and Gary A. Shaw, who at various times and in various combinations have contributed significantly to this project. In particular, Steve initially defined the required algorithms, Joe developed the original architecture and provided design guidance, and Gary developed the revised I/Q filter structure and contributed to the TDM bus design.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

LIST OF TABLES

1. INTRODUCTION

This report is the first of several that will document the radar signal processor test bed developed at Lincoln Laboratory in Group 27. This effort represents a demonstration of the radar signal-processing techniques put forth in an earlier report [1], as well as the compact and fault-tolerant architectures that might be required for a variety of on-board signal processors. Plans to integrate the signal processor test bed with a radar receiver test bed and phased-array antenna at the Near Field Test Facility will round out the complete system demonstration.

The architecture for the signal processor test bed is derived primarily from a previous report [1] that presented a set of digital signal-processing algorithms for an adaptive radar and then described a possible implementation, given a set of baseline radar parameters. After refinements, the resulting architecture has been partitioned into three major subsystems: the front end, the adaptive nulling processor, and the vector processors. Figure 1 presents a functional block diagram of the system. The blocks labeled *I/Q*, *EQ*, *PC*, and *Beamformer* constitute the front end subsystem, which is partitioned into eight identical modules that have been fabricated on VMEbus standard 9U×220-mm wirewrap boards. The adaptive nulling processor is a single circuit board that performs adaptive nulling computations and acts as the system host. This board resides in the same card cage as the front end modules. The test bed also includes four identical vector processors, each composed of two 9U×220-mm boards. The vector processors are programmable and perform Doppler processing along with other tasks. A system clock board (not shown in the block diagram) also resides in the front end subsystem card cage and rounds out the hardware for the signal processor test bed.

The front end subsystem receives eight channels of sampled data from the eight radar receiver A/D modules, each at a rate of 4 million samples per second. It processes this data with a set of programmable coefficient finite impulse response (FIR) filters. The first set of filters performs inphase and quadrature signal separation (demodulation) and then downsamples the data streams by three. The next set of filters performs the tasks of channel equalization (to correct for receiver channel mismatch) and pulse compression. The coefficients for these filters are loaded by the adaptive nulling processor via a VMEbus compatible interface. The data stream from the eight sets of filters is then broadcast over a time-division-multiplexed (TDM) bus to the four beamformers and the adaptive nulling processor. Each beamformer subsamples the data by two and then creates a weighted sum of the eight complex data streams. The adaptive nulling processor also reads data from the TDM bus and uses this data to compute the adaptive nulling weights. These weights are downloaded to the beamformers via the VMEbus compatible interface. After beamforming, the resulting complex data streams are distributed along four byte-wide buses to each of the four vector processors. Algorithms for clutter cancellation, Doppler processing, and detection are implemented in these programmable processors.

The remainder of this report will document in detail the eight-module front end subsystem. References to other portions of the test bed will be limited to functions and features that directly

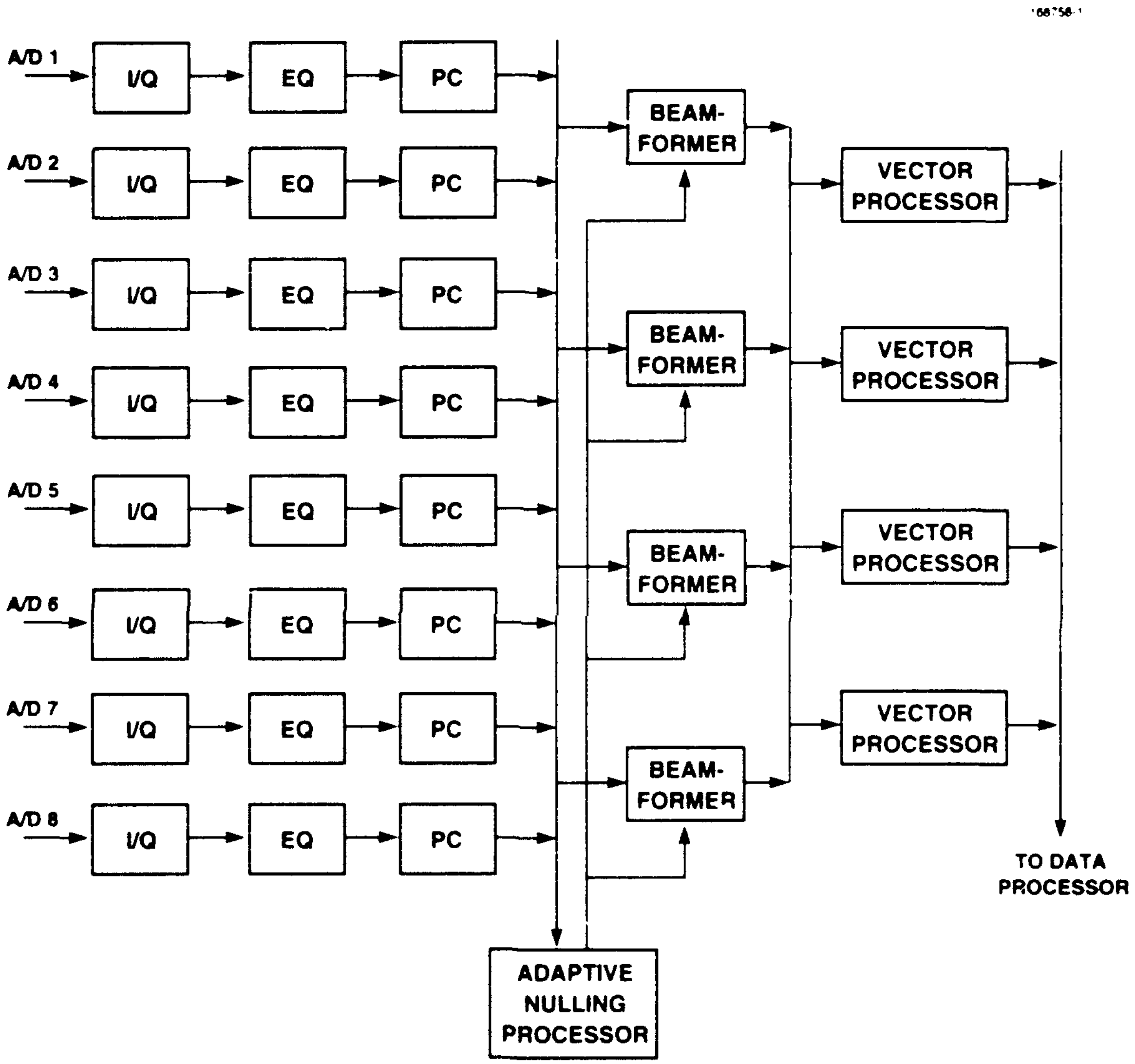


Figure 1. Block diagram of the processor test bed.

affect operation of the front end. Section 2 describes the hardware design of the front end, with special attention given to the interfaces to other systems and subsystems. Section 3 describes the front end from the programmer's point of view. Detailed descriptions of the filter coefficient placement and the various control options will be presented. Section 4 addresses miscellaneous front end subsystem issues, including a description of the hardware and requirements of the system clock board. Appendix A documents the radiation tolerance tests that were performed locally on the Inmos A100. Issues pertaining to a next-generation system, including compact packaging and design enhancements, are covered in Appendix B.

2. THE FRONT END SUBSYSTEM

The front end subsystem has been partitioned into eight identical front end modules that have been constructed on VMEbus standard [2] 9U× 220-mm wirewrap boards. The block diagram of the entire processor test bed is reshown in Figure 2. The blocks labeled *I/Q*, *EQ*, *PC*, and *Beamformer* constitute the front end subsystem. This version of the block diagram depicts a region outlined by a dashed line, which contains one of the eight modules.

Each module interfaces to one A/D channel and performs filtering for that channel. Each module outputs data during one time slot on the time-division-multiplexed (TDM) bus and receives data from all eight time slots. And finally, each module contains the logic for one-half of a digital beamformer. That is, either the inphase or quadrature portion of the weighted sum is formed on one module, and the corresponding component is formed on the adjacent module. Thus, eight identical modules form eight sets of front end filters and four complete beamformers. If one module fails, the radar loses one A/D channel and one beamformer and becomes limited to a subset of its operating algorithms. Schemes for including a ninth module to allow for sparing have been studied but are not included in this test bed. A block diagram of the data flow in one of the front end modules is shown in Figure 3.

Section 2.1 describes in detail the interface between the front end modules and the radar receiver A/D modules. Section 2.2 covers the input multiplexing of the data and the FIR filters. The details of the TDM bus are presented in Section 2.3 and the beamformers in Section 2.4. The beamformer output bus and its associated timing are defined in Section 2.5. And finally, Section 2.6 covers miscellaneous hardware features of the front end module.

The eight-module subsystem resides physically in a single card cage along with the adaptive nulling processor and a system clock board. The ten boards are situated physically in the rack as shown in Figure 4.

The adaptive nulling processor is the sole bus master for the backplane VMEbus. It is tasked with programming the front end modules as well as sampling data on the TDM bus. The system clock board provides a 24-MHz differential clock to the front end modules. In addition, it receives and synchronizes timing signals from the radar receiver. This function will be described in greater detail in the next section.

2.1 A/D Interface

The processor test bed has been designed to interface to the A/D converters of an eight-channel radar receiver. A compatible system has been developed at Lincoln Laboratory and will be used for experiments at the Near Field Test Facility (also being constructed at Lincoln Laboratory). In addition, testing of the front end, as well as the completed signal processor test bed, can be performed via a Test Vector Generator (TVG) system [3] that emulates the radar receiver. Both the front end and the TVG system have been designed to accommodate 16-bit data samples. However,

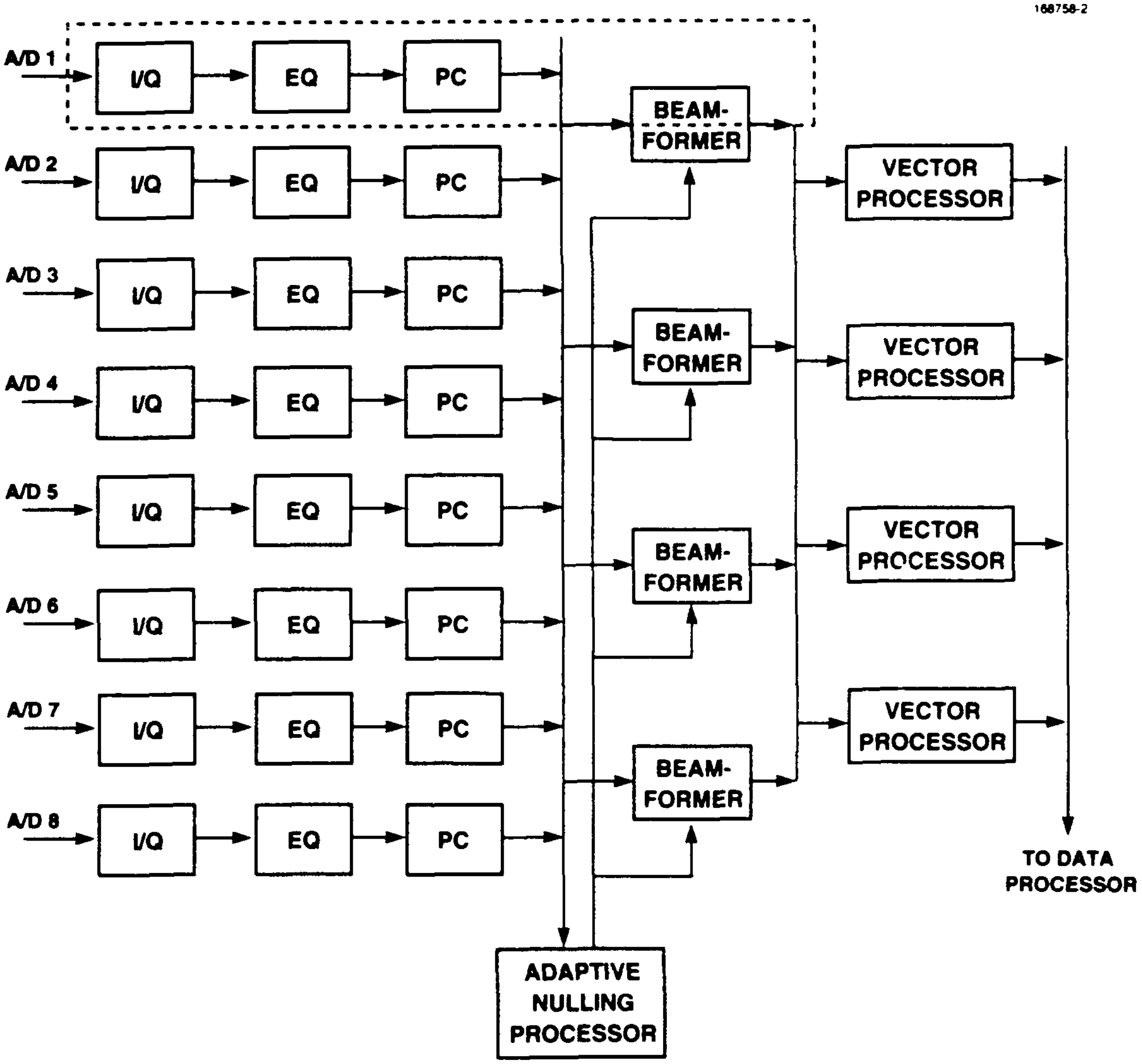


Figure 2. Block diagram of the processor test bed with front end module partition.

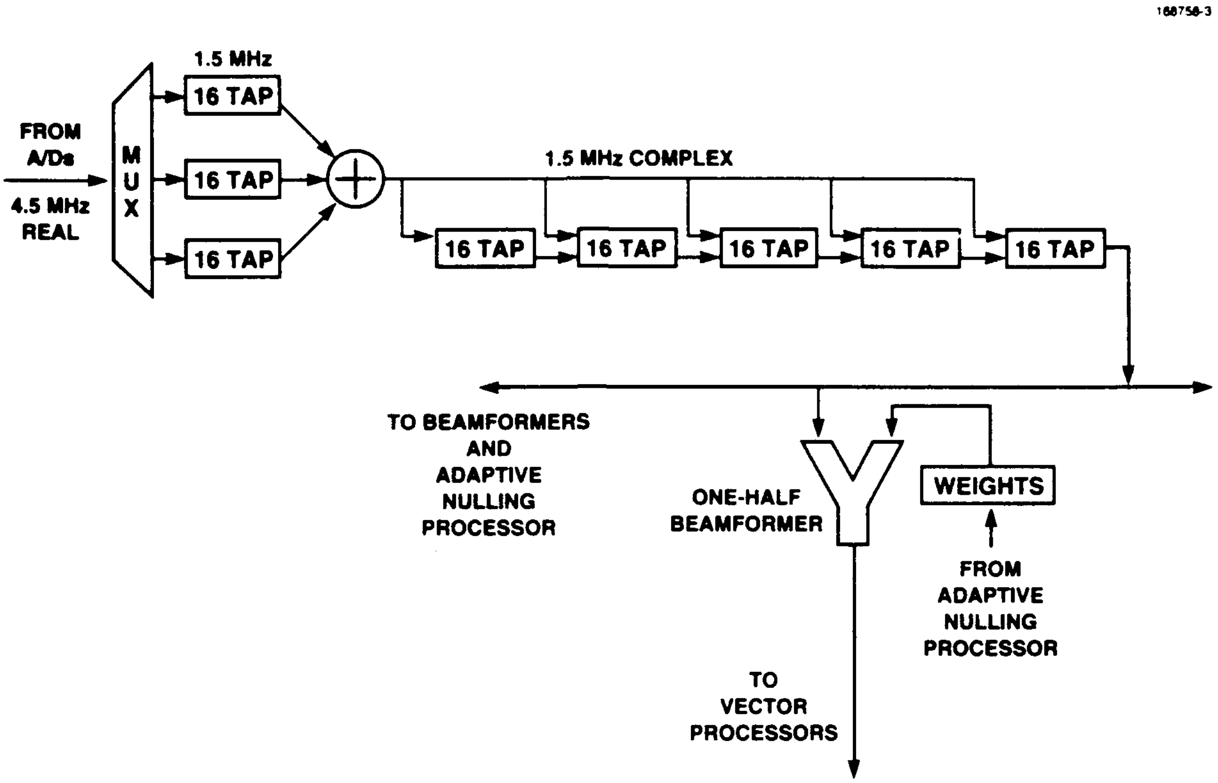


Figure 3. Block diagram of the front end module data flow.

the current A/D technology in the radar receiver limits word length from 12 to 14 bits. When the front end is interfaced to the radar receiver, the data word will be sign extended on the most significant bits (MSBs), filled with zeros on the least significant bits (LSBs), or a combination of both. The following section presents the mechanical, electrical, and timing considerations of the general-purpose 16-bit interface. Wherever applicable, the differences in interfacing to the radar receiver will be addressed.

2.1.1 Hardware Interface

The front end subsystem has been designed to accept 16-bit digitized data, sampled at a rate of fewer than or equal to 4.5 million samples per second on each of eight channels. The data is received in a bit-by-bit, channel-by-channel, parallel fashion. Each A/D channel is connected to the processor test bed with a twisted-pair ribbon cable. At the transmit end (the A/D converters in the radar receiver or the TVG output ports), each of the 128 data lines (96 for the radar receiver) uses individual drivers and is terminated with a matched series resistor. At the receive end (the

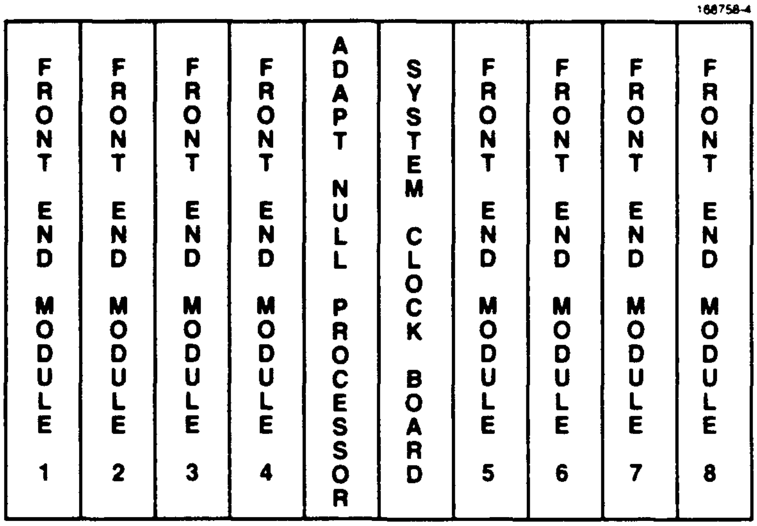


Figure 4. Front end subsystem card cage.

processor test bed), each of the data lines has an individual line receiver, but no termination is required. (Laboratory experimentation has indicated that an 82-ohm series termination resistor at the source, with no termination at the destination, successfully removes the effects of ringing and reflection from the twisted-pair ribbon cables being used. See Figure 5 for circuit description.) In each twisted pair, one of the wires is used to carry the data bit, and the other is grounded at both ends.

Each channel requires a 34-wire (17 signal) twisted-pair cable. The front end subsystem and the TVG system are each equipped with a panel that contains eight 34-pin ribbon cable headers. Each connector is labeled with the corresponding A/D channel. The pin assignment for these connectors is presented in Table 1.

The eight radar receiver channels employ 12-bit A/D converters, operating at 4.5 million conversions per second, to digitize the incoming waveform. The front panel on the radar receiver provides 26-pin ribbon cable headers rather than the 34-pin headers on the processor test bed. The pin designation for these headers is presented in Table 2. When interfacing to the radar receiver, it will be necessary to sign extend the MSBs of the 12-bit samples, zero fill the LSBs, or some combination of both. These connections can be made in the cables, connectors, or front end subsystem wiring harness.

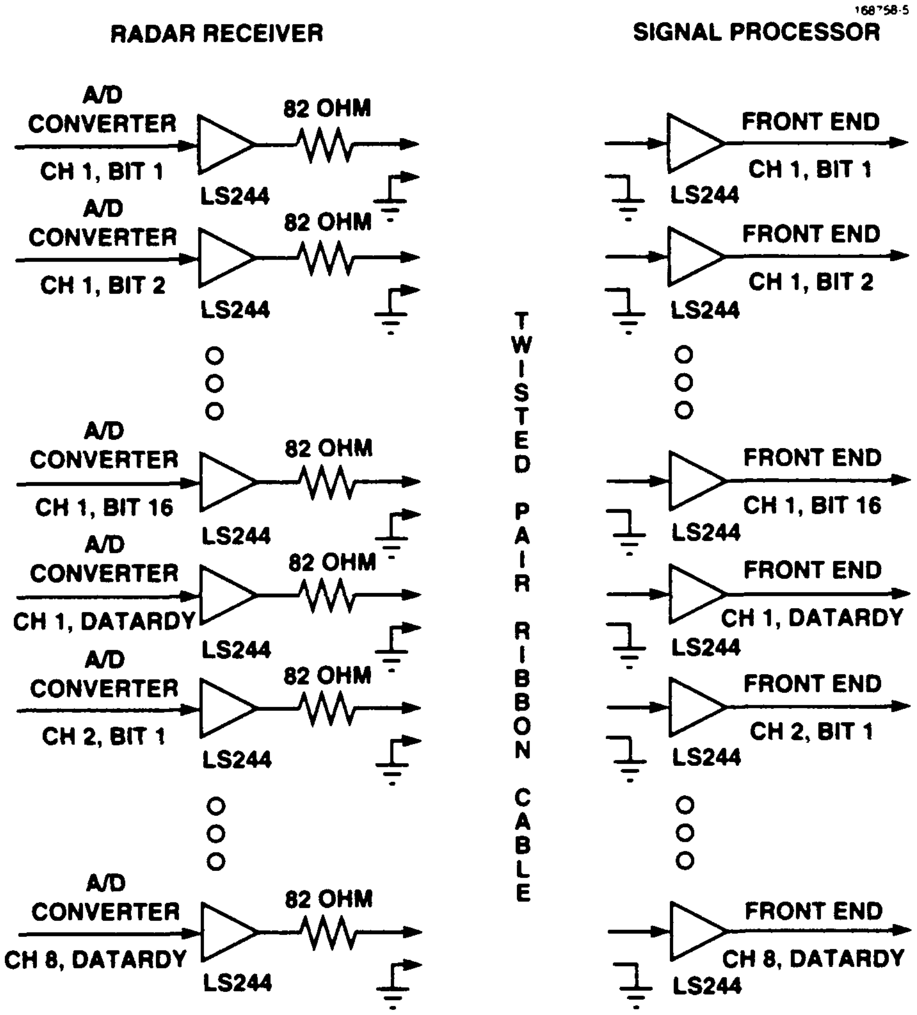


Figure 5. Hardware interface diagram.

TABLE 1

Pin Assignment for Test Vector Generator to Front End Subsystem Interconnect

Pin	Signal	Pin	Signal
1	DAT00 (LSB)	18	DAT08 RTN
2	DAT00 RTN	19	DAT09
3	DAT01	20	DAT09 RTN
4	DAT01 RTN	21	DAT10
5	DAT02	22	DAT10 RTN
6	DAT02 RTN	23	DAT11
7	DAT03	24	DAT11 RTN
8	DAT03 RTN	25	DAT12
9	DAT04	26	DAT12 RTN
10	DAT04 RTN	27	DAT13
11	DAT05	28	DAT13 RTN
12	DAT05 RTN	29	DAT14
13	DAT06	30	DAT14 RTN
14	DAT06 RTN	31	DAT15 (MSB)
15	DAT07	32	DAT15 RTN
16	DAT07 RTN	33	Encode
17	DAT08	34	Encode RTN

TABLE 2

Pin Assignment for Radar Receiver Connectors

Pin	Signal	Pin	Signal
1	DAT00 (LSB)	14	DAT06 RTN
2	DAT00 RTN	15	DAT07
3	DAT01	16	DAT07 RTN
4	DAT01 RTN	17	DAT08
5	DAT02	18	DAT08 RTN
6	DAT02 RTN	19	DAT09
7	DAT03	20	DAT09 RTN
8	DAT03 RTN	21	DAT10
9	DAT04	22	DAT10 RTN
10	DAT04 RTN	23	DAT11 (MSB)
11	DAT05	24	DAT11 RTN
12	DAT05 RTN	25	Encode
13	DAT06	26	Encode RTN

2.1.2 Data Transfer Timing

The front end subsystem also requires a "data ready" signal called Datardy, which is used to latch the successive valid A/D samples. The Datardy signal is normally at logic level "0" (0 volts) and transitions to logic level "1" (5 volts) at the center of the "output data valid" interval of the A/D converter. The Datardy signal is actually a "processed" version of the Strobe signal transmitted by the TVG or radar receiver. It is critical to the front end timing that the eight Datardy signals are synchronized to the 24-MHz clock cycle. Therefore, one of the miscellaneous functions of the system clock board is to receive all eight copies of the data Strobe signal, select one, synchronize it with the 24-MHz system clock, and drive this Datardy signal down the backplane to all eight front end modules. Both the clock and Datardy signals are driven and received differentially using the AT&T 41 series drivers and receivers. The Strobe signal for each channel is transmitted and received along the twisted-pair ribbon cables using the same techniques that were described for the data lines.

It is a requirement of the transmitting system (either the TVG or the radar receiver) that the eight data strobes be sufficiently close in timing so that any one of the eight strobes can be used to latch data by all eight front end modules. The data strobes are combined to guarantee that the eight filter modules remain in lock step. That is, because the Datardy signal is synchronized to the local 24-MHz clock by the clock board, each module receives data and initiates filter computation on the same clock cycle. The disadvantage to this technique is in the fault tolerance of the system. Allowances should be made for the failure of one of the eight A/D modules while still maintaining a subset of the radar operation. This could be accomplished by a programmable switching device that would allow the clock board to choose which of the eight Encode signals to broadcast. For this reason, all eight Encode signals are brought to the clock board where one is synchronized to 24 MHz and broadcast as Datardy. (Other options are discussed in Section 4.3.) In the current implementation of the system clock board, this decision is implemented trivially by simply selecting the channel 1 strobe. In a deployed version of the system, this decision should be made according to some fault-tolerant scheme that can accommodate single or even double A/D module failures without losing the Datardy signal.

When interfacing the front end to the radar receiver, the required Datardy signal can be derived from the A/D converter Encode pulse. Thus, the radar receiver transmits its A/D Encode pulses in addition to the 12-bit data for each channel. The Encode pulse for the A/D converter is a normally low signal that transitions to a high logic level to initiate data conversion. The output of the converter can then be captured at its return transition to a low level. The output data that is latched at the Nth high-to-l0 transition of the Encode signal corresponds to data that was sampled at the $(N-2)$th low-to-high transition of this signal. Thus, in deriving the Datardy signal, the front end subsystem will ignore the first two high-to-low transitions of the Encode pulse at the beginning of each Pulse Repetition Interval (PRI). (See Figure 6.) The width of the Encode pulse is one-half period of the 4.5-MHz sampling clock.

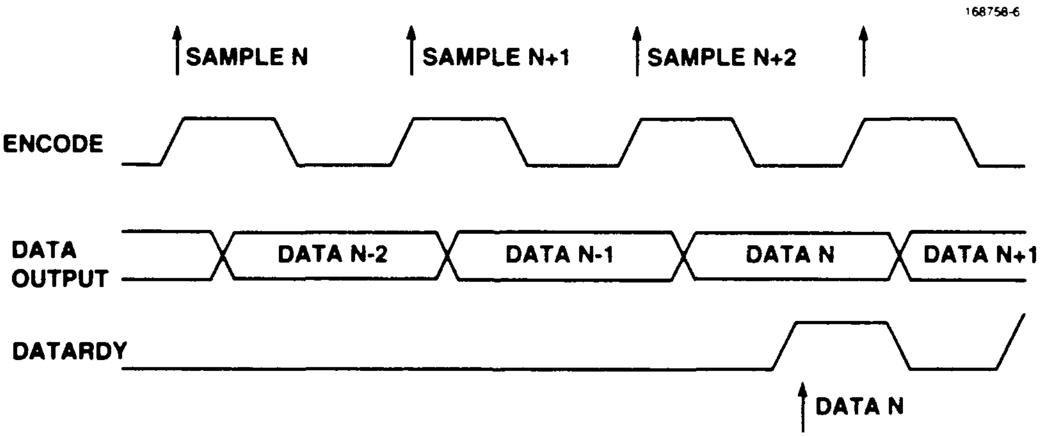


Figure 6. Deriving Datardy from the A/D Encode pulse.

2.1.3 Pulse Repetition Interval Timing

One other timing signal is required for operation of the signal processor. The "PRI pulse," which is used to demarcate the contiguous Pulse Repetition Intervals, synchronizes the incoming data with the transmit intervals of the radar. This signal line is normally at a logic level "0" and pulses to a logic level "1" for a duration not fewer than 100 ns between PRI periods. The Strobe (or Encode) pulses from the A/D channels are suspended a minimum of 100 ns before, during, and 100 ns after the occurrence of the PRI pulse. Thus, the Strobe signals are suspended for a minimum of 300 ns. The PRI pulse is transmitted and received using the same matched resistor technique described for the data transmission. This signal requires a ninth twisted-pair cable. Thus, in addition to the eight 34-pin headers installed on the system interface panel, an additional 16-pin header has been installed and labeled "PRI." The PRI signal is located on pin 1 of this connector with pin 2 used as its return.

The front end subsystem receives the A/D samples and begins processing this data with a complex digital finite impulse response (FIR) filter. In interpreting the output results of these filters it is desirable to have the filters in an a priori known state at the beginning of each PRI when interpreting output results. The simplest solution would be to asynchronously clear the filter taps during the PRI pulse. However, the FIR filter chips used by the front end modules do not support this function. Instead, the current approach is to force the input data to zero for a sufficient number of Datardy strobes to clear out the filter cascade. Thus, after the occurrence of the PRI pulse, the front end receives a specified number of valid A/D samples (currently limited to a maximum of approximately 850 samples by the vector processor memory). For each subsequent Datardy pulse within the PRI interval, the front end zeros out its input data. A maximum of 288 extra pulses will be required to completely clear the filter cascade. This process is restarted at the occurrence of the next PRI pulse. The timing diagram of Figure 7 presents the relationship between the PRI pulse and the Datardy pulses. The time period T refers to the duration of the PRI pulse and the length of time on either side of the PRI pulse in which there are no Datardy pulses. These time periods need not be identical but must be 100 ns or more.

This process of clearing the front end filters would not be a requirement of an operational system. It is merely a convenience for testing purposes because it enables us to generate, bit for bit, a priori known data at the output of the filters. The front end can be operated without the extra strobes, but this will result in transient effects at the beginning of each PRI induced by data from the previous PRI.

2.2 FIR Filters

The major computational task of the front end subsystem is to digitally filter the data streams as received from the radar receiver. The design of the digital filters is based on a commercially available integrated circuit, the Inmos A100. This device, packaged in a 10 × 10 pin grid array (PGA), implements a 32-tap finite impulse response (FIR) filter. A user's model block diagram of the A100 is presented in Figure 8. Understanding the functionality of this device is integral to

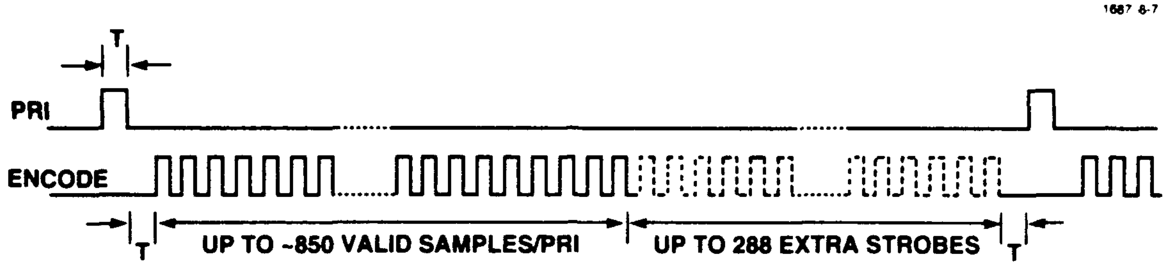


Figure 7. PRI and Encode pulse timing.

understanding the front end filter design. While some explanation of the A100 features is included in this report, more complete information is in the Inmos literature [4].

The data stream can be input to the A100 via its memory interface by writing to the Data Input Register (DIR). This approach simplifies the hardware design but artificially limits the throughput of the device to the bandwidth of the memory interface. Data can also be input to the A100 via a dedicated input port. Likewise, the filtered data stream can be read out of the A100 from the Data Output Low (DOL) and Data Output High (DOH) registers but can be more efficiently taken directly from the data output port. The front end design uses the dedicated ports for both input and output. The cascade input is designed to allow seamless cascading of multiple A100s to form longer length FIR filters. This feature is used in the equalization and pulse compression filters.

The A100 implements a modified version of the canonical FIR filter structure. The typical FIR filter structure passes the input data stream through a pipelined set of unit delay registers. At each clock cycle, a weighted sum of the elements from each delay interval is formed. By contrast, the A100 premultiplies each input sample by all of the FIR coefficients. The products are then summed along a registered delay line. The modified architecture, implemented by the A100, produces the same numerical result. The two alternate architectures are demonstrated in Figure 9.

The filter coefficient registers of the A100s are mapped into the VMEbus memory space and are loaded by the adaptive nulling processor. The A100 includes 32 16-bit Current Coefficient Registers (CCRs) and 32 16-bit Update Coefficient Registers (UCRs). The filters always operate with data from the CCRs, but coefficients can be swapped between the corresponding CCRs and UCRs in one clock cycle simply by writing to a bit in one of the A100 control registers. Additionally, the A100 can be configured in a "continuous swapping" mode where the contents of the CCRs and UCRs are swapped after each data sample is input. This unique feature enables the A100 to implement a 16-tap complex filter rather than a 32-tap real filter [5]. The front end uses this complex filter configuration to demodulate the real data stream coming from the radar receiver A/D modules into its inphase and quadrature components. The resulting complex data stream is

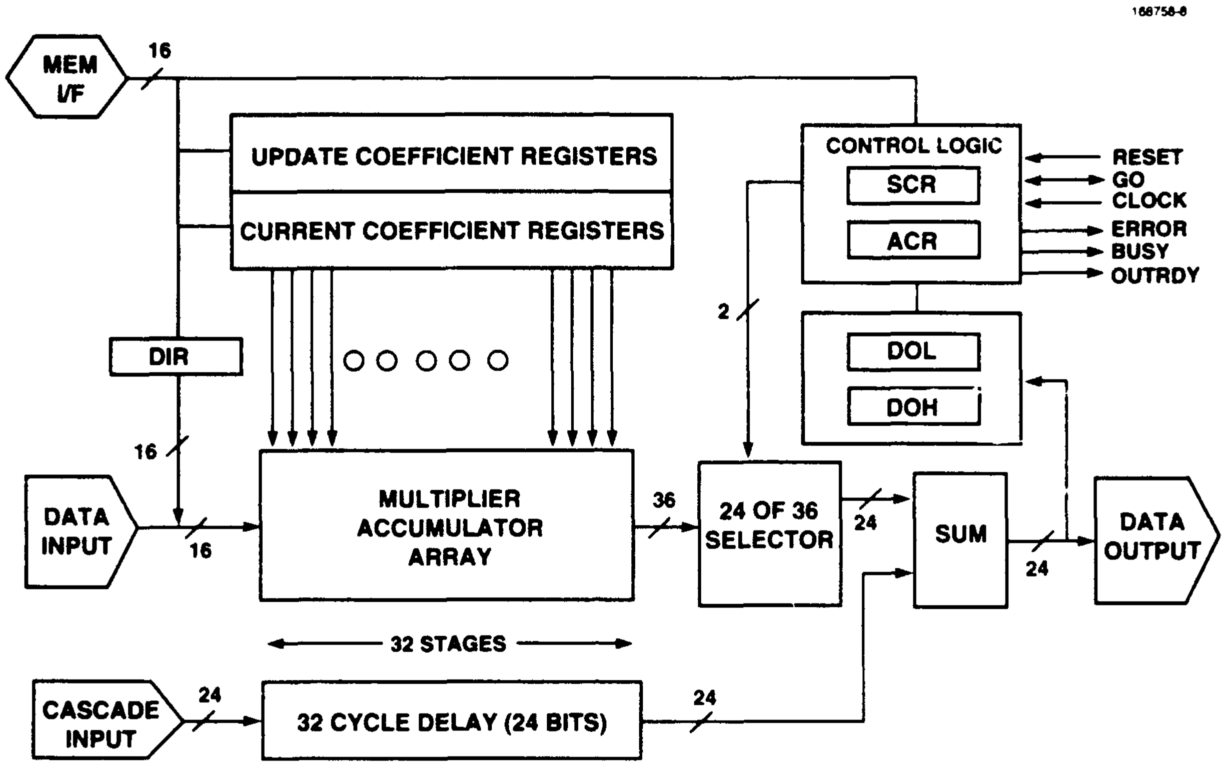


Figure 8. User's model of the Inmos A100.

then processed by another set of complex FIR filters that provide channel equalization and pulse compression.

The A100 accepts a 16-bit input data stream and filters it with 4-, 8-, 12-, or 16-bit coefficients. The maximum throughput, in samples per second, is a function of the coefficient word length according to the equation

$$THROUGHPUT = \frac{2 \times F}{L_C} \quad ,$$

where F is the clock frequency, and L_C is the length of the coefficient in bits. Inmos sells a variety of A100s that support different maximum clock rates. The front end was built with 30-MHz A100s that, for historical reasons, we are operating at 24 MHz. The algorithms [1] employed by the signal processor test bed require 16-bit coefficients to meet the signal-to-noise ratio goals of the system. This requirement yields a maximum input data rate of 3 MHz.

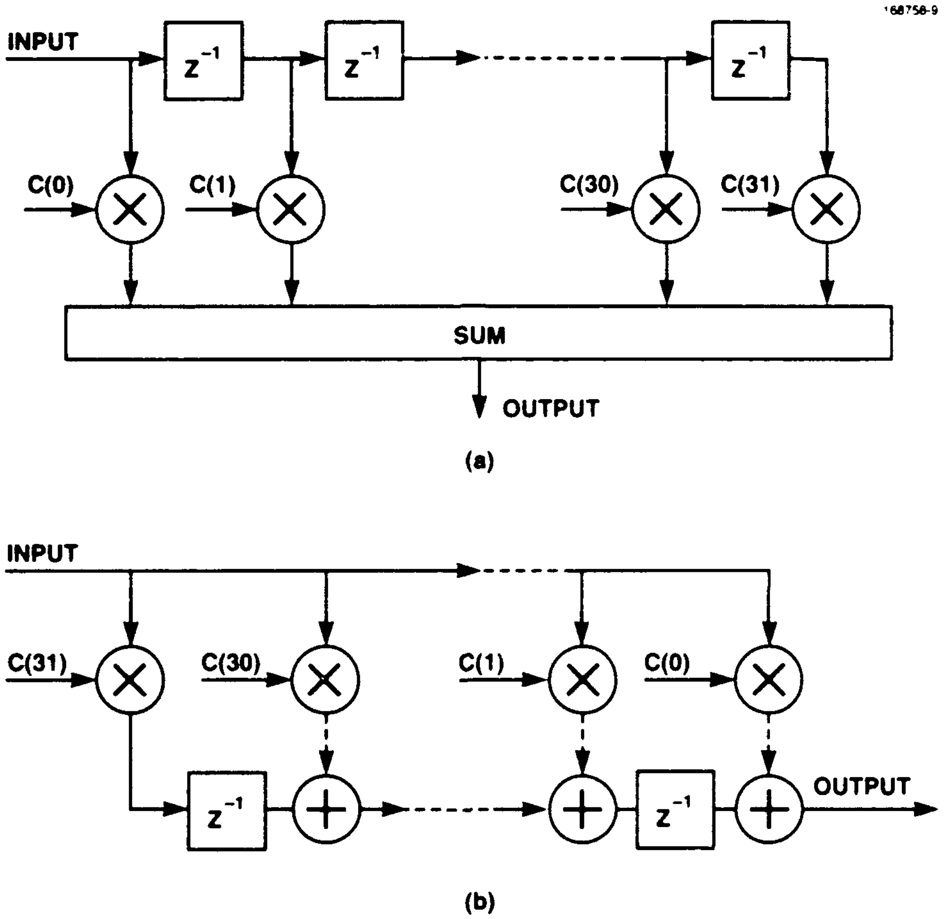


Figure 9. FIR filter structures: (a) canonical FIR filter and (b) modified A100 structure.

The Inmos A100 is clearly a unique and integral part of the front end subsystem. The radiation tolerance of the components may be of concern in some potential applications of the signal processor. Therefore, before the system design began, radiation tolerance tests of the A100 were performed. The test results indicate that the A100 can be expected to tolerate a total dose exposure of at least 10 krad(Si). These tests are documented in Appendix A.

2.2.1 Inphase and Quadrature Filters

The task of I (inphase, real) and Q (quadrature, imaginary) signal separation is performed digitally in the front end subsystem. The incoming data stream, 16-bit real data at 4.5 MHz, is filtered with a complex FIR filter. The real signal has a center frequency of 1.5 MHz. This ratio of sampling rate to center frequency allows the downsampling of complex data stream by three to effect a frequency shift down to baseband [1]. Thus, the output rate of the I/Q filters is 1.5 million complex samples per second or 3.0 million 16-bit words per second.

Because of the 24-MHz system clock and the requirement for 16-bit coefficients, the maximum input data rate of the A100s is 3 MHz. This rate is reduced to 1.5 MHz in order to implement a complex filter. A single A100 (or linear cascade of A100s) will therefore not suffice in filtering the A/D data stream. However, we can take advantage of the fact that the filtered I/Q data stream will be downsampled by three and only compute every third sample. This approach is accomplished by applying the data stream to three A100s in an interleaved manner and then recombining (summing) the output data streams. A block diagram of this portion of the module is shown in Figure 10.

The 16-bit data arrives from the radar receiver along twisted-pair ribbon cables that are terminated at the transmit end. The data is buffered by the front end modules with a pair of octal buffer/line drivers (74LS244). The data is then distributed to each of three 16-bit double-buffered registers. The first half of these double buffers is constructed from two octal D flip-flops (74ACT374) with the outputs permanently enabled. The second half consists of another pair of octal D flip-flops (74ACT273), which include an asynchronous clear function. (This feature can be used to zero out the input data stream when clearing out the filters.)

The control logic includes a state-machine with three states that point to each of the three buffers in turn. Initially, the state-machine points to the top buffer, and the first data sample that arrives is clocked into the front half of the top buffer. The next data sample is clocked into the front half of the middle buffer, and the third sample into the front half of the bottom buffer. After the third sample has been latched, a second state-machine is triggered, and it sequences through the data input cycle. First, all three samples are clocked simultaneously into the back half of the double buffers. (This frees up the front half to start collecting the next three samples.) Next, the GO signal is asserted on all three A100s. On the first rising edge of the system clock after the assertion of GO, the A100s read in the three data samples and begin filtering this data with the inphase (real) half of the I/Q filter. When the A100 is ready, the state-machine reasserts the GO signal, and the same real data sample is reread and filtered with the quadrature (imaginary) half of the I/Q filter. By the time this process has been completed, three new data samples have been

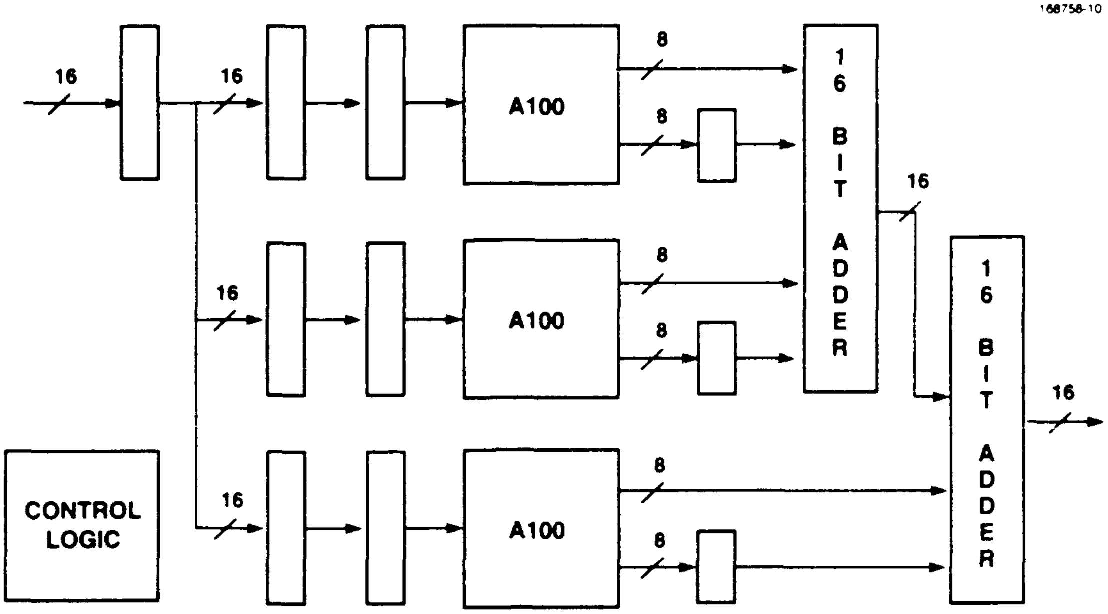


Figure 10. I/Q filter portion of the front end module.

loaded into the front half of the double buffers and are ready to be transferred over to the back half. Note that the 4.5-MHz input rate is reduced to 1.5 MHz by the interleaving/downsampling and then doubled to 3.0 MHz by clocking each sample into the filters twice. This rate is exactly matched to the 3.0-MHz maximum input rate of the A100s. If the input data rate exceeds 4.5 MHz, trios of input samples will be dropped at random. If the input data rate is fewer than 4.5 MHz, the filters will continue to operate at full speed with "dead" time inserted between samples. The output data will be completely accurate.

It is absolutely imperative that the three A100s on all eight front end modules clock in the data samples on the same rising edge of the 24-MHz system clock.[1] This timing requirement is the reason that the Datardy signal, which is used to input data to all eight modules, is synchronized to the 24-MHz clock at the system clock board and then distributed to the eight front end modules.

The three A100s, each implementing a 16-tap complex filter, yield an effective 48-tap complex filter with 3X downsampling. The coefficients for this filter are interleaved among the three A100s. The real half of the coefficients are placed in the CCRs, and the imaginary half are placed in

[1] This requirement will become clear after understanding the TDM bus timing presented in Section 2.3.

the UCRs; the banks are swapped on each cycle. Figure 11 shows the arrangement of the filter coefficients and the distribution of the incoming data. The figure is adapted in part from the Inmos literature [5]. The figure shows three five-tap versions of the A100 implementing a nine-tap complex filter that downsamples by three. The A100 implements a modified filter architecture. That is, each data sample is multiplied by all coefficients as it arrives, and the products are delayed and summed through a pipeline structure. Figure 11 can be extended to 32 taps per filter block to yield the actual front end I/Q filter structure. The result is that, at each output cycle, the three A100s output the alternating inphase and quadrature components of the filtered signal as three partial sums.

Two 16-bit arithmetic logic units (ALUs), Logic Devices Inc. part number L4C381, are used to combine the partial sums. The ALUs are configured as asynchronous adders. The first ALU adds the output samples from the upper two A100s. The second ALU adds this partial sum to the output from the third A100. The net asynchronous time delay of the two adders is much less than the cycle time of the A100. Thus, we are able to combine the three streams of data through these devices without latching the results. Rather, the resultant I and Q data are piped into the next stage of filters.

To avoid overflow problems, the A100 performs all internal calculations with 36 bits and then allows the user to select a 24-bit output field from the internal result. The output data bus is actually 12 bits wide, and the upper and lower half of the filter data are output successively. The front end module must maintain the data width of 16 bits for future operations. The hardware is designed to extract the center 16 bits from the 24-bit output word. This result is accomplished by latching the upper 8 bits of the 12-bit bus when the least significant half of the data is output and then using (but not latching) the lower 8 bits when the most significant half is output. (The filter coefficients are calculated carefully to provide the necessary gain to scale the valid output data to lie in the center 16 bits of the 24-bit range of the A100 bus.)

The A100 provides a signal called Outrdy, which can be used to latch the output data. The intended approach is to use the falling edge of Outrdy to capture the least significant 12 bits and the ensuing rising edge of Outrdy to capture the most significant 12 bits. However, the Outrdy signal is apparently generated by the A100 in some asynchronous fashion, and so Inmos cannot guarantee between which edges of the 24-MHz system clock the Outrdy edges will occur. Because the front end subsystem absolutely requires that all data between the eight modules are latched on the same 24-MHz clock edge, an alternate approach was devised. Inmos *can* guarantee that, given a particular coefficient word length, the upper and lower portions of the output word can be latched a predetermined number of clock cycles after the assertion of the GO signal. Thus, a 40-bit shift register was added to the front end control logic. The GO signal is input to the shift register as well as the A100s. Fifteen clock cycles later the least significant portion of the output word is latched. Four clock cycles are given to allow the most significant 12-bit output to settle and for the partial sums to propagate through the adders. Thus, 19 clock cycles after the initial GO signal, a second GO signal, called GOPC, is sent to the cascade of A100s implementing the equalization and pulse compression filter.

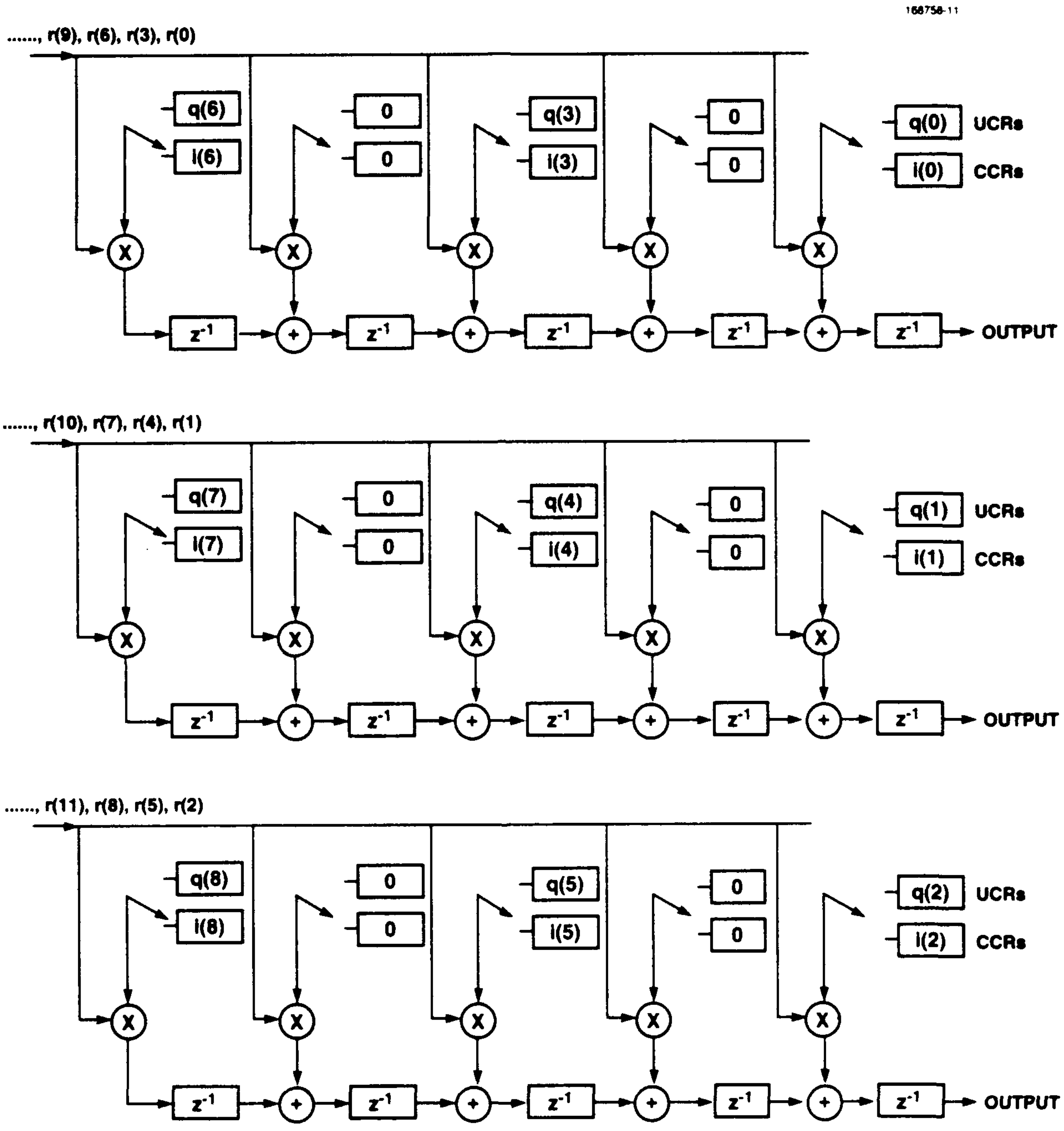


Figure 11. I/Q filter coefficient arrangement.

Thus, the final inphase and quadrature data stream is passed on to the next stage of front end filtering. This data stream contains 16-bit real data and 16-bit imaginary data transmitted at a rate of 1.5 million complex samples per second. The data represents the original A/D signal demodulated into its complex inphase and quadrature components and frequency shifted to baseband.

2.2.2 Equalization and Pulse Compression Filters

After I/Q filtering, the next two tasks of the front end module filters are to perform pulse compression and channel equalization filtering. These two functions are disjoint in nature, and their filter coefficients are computed separately; however, they are performed together in the same FIR filter chain. The pulse compression filter coefficients are predetermined by the modulation of the transmit waveform. The channel equalization filter coefficients are computed adaptively [1]. The two filter functions are convolved by the adaptive nulling processor to form a single function and downloaded to the A100s via the VMEbus interface. The chain of five A100s implements an 80-tap[2] complex FIR filter. Thus, the total filter length of the two functions must not exceed 80.[3] A block diagram of this portion of the front end module is given in Figure 12.

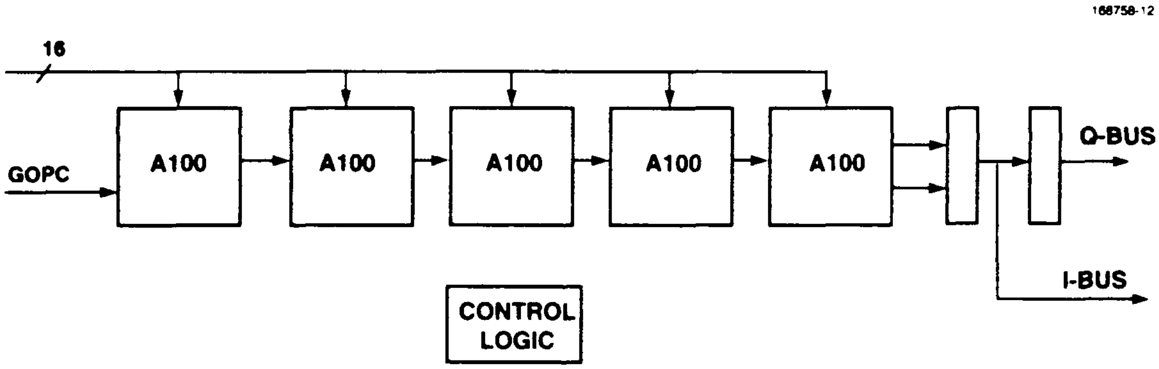


Figure 12. Pulse compression and channel equalization filters.

[2] Actually, in configuring the complex filter coefficients, one tap in the cascade is lost. Although the filter does contain 160 real taps, only a 79-tap complex filter can be implemented.

[3] Convolving an m-tap filter with an n-tap filter results in a $(m + n - 1)$-tap filter. Because we can only implement 79 complex filter taps, $m + n \leq 80$.

The A100s output 24-bit data in two 12-bit halves, and the front end uses the middle 16 bits of this 24-bit output word. Two octal D flip-flops (74ACT374) are used to latch the lower and then upper halves of the desired 16-bit output words. The real and imaginary components of the data stream are output in successive cycles of the A100. The front end module uses a time-division-multiplexed (TDM) bus to broadcast the filtered data from all eight modules to the beamformers and the adaptive nulling processor. This bus contains an inphase half (I-bus) and a quadrature half (Q-bus) that transmit simultaneously. Therefore, a second set of buffers is appended onto the first set to allow the filter output to store up a full 32-bit-wide data sample for transmission. This process is described in greater detail in the next section. The 40-bit shift register, described in the I/Q filter section, is used as the control logic for the equalization and pulse compression filters as well. This shift register generates the GOPC signal used to input data into the 80-tap filter cascade, as well as the timing signals used to clock the lower and upper portions of the filter result out of the cascade. Some additional control logic is implemented to queue up the inphase and quadrature data and then to load it into the TDM bus drivers.

As with the I/Q filters, the channel equalization and pulse compression filters implement a complex FIR filter. However, the difference between these two structures is that the I/Q filters input real data and the equalization and pulse compression filters input complex data. Once again, the bank swapping feature of the A100 is used to implement the filter, but a more complicated approach to loading the filter coefficients is required. Figure 13, modified from the Inmos documentation [5], demonstrates the required configuration.

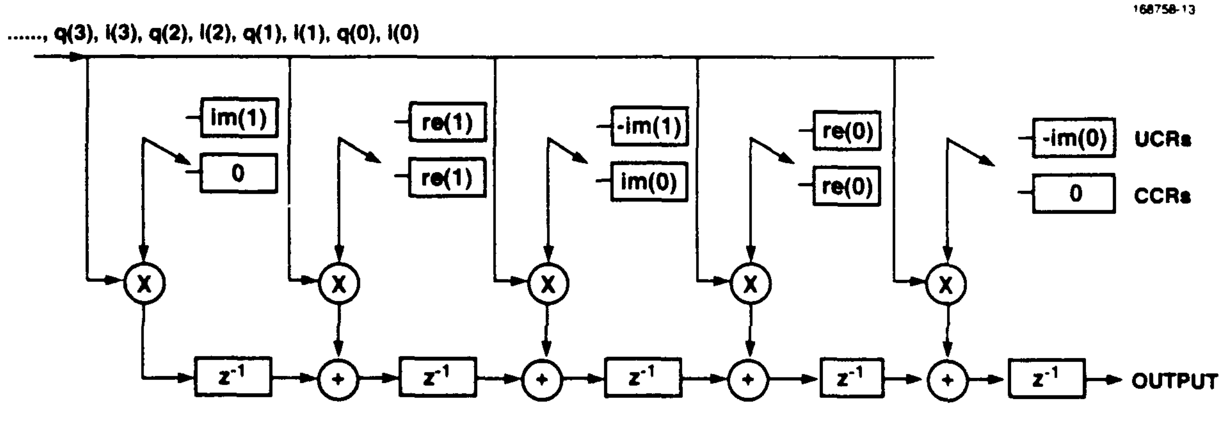


Figure 13. Pulse compression and channel equalization filter coefficient arrangement.

The input data is labeled $i(n)$ and $q(n)$ for inphase and quadrature, and the filter coefficients are labeled $re(n)$ and $im(n)$ for real and imaginary. For the purposes of this report, these two notations are actually interchangeable. Swapping these CCR and UCR contents on each cycle results in an interleaved complex FIR filter.

The cascade of five A100s is a seamless implementation of Figure 13 extended to 160 taps (yielding 80 complex taps). It should be noted here that the A100 farthest on the right contains the lowest order coefficients, and the A100 farthest on the left contains the highest order coefficients. However, in the VMEbus memory map, the former A100 is designated as A100 number 7 and the latter A100 is designated as A100 number 3. This nonintuitive mapping can be corrected in the software written for the host processor.

The output of these filters is passed onto the TDM bus for broadcast to the other seven front end modules, as well as the adaptive nulling processor. The output is a complex data stream, with 16-bit real and imaginary components, operating at a rate of 1.5 million complex samples per second. The baseband I and Q data have been pulse compressed (matched filtered with the transmitted pulse), and any channel mismatch in the analog radar receiver has been equalized. The next step in the processing will be to form radar "beams" by computing a weighted sum of the eight channel data on a sample-by-sample basis.

2.3 Time-Division-Multiplexed Bus

The backbone of the front end subsystem architecture is the time-division-multiplexed (TDM) bus, which is used to broadcast filtered data from each of the eight receiver channels to each of the four digital beamformers and the adaptive nulling processor. When the front end filters are receiving sampled data at a 4.5-MHz rate, they are outputting complex data (i.e., a 16-bit inphase component and a 16-bit quadrature component) at a rate of 1.5 MHz. Each of the four beamformers will downsample the data stream by two and then compute a weighted sum of the eight resulting data streams. (The reduction in sample rate is possible due to the low-pass characteristic of the pulse compression filter.) The adaptive nulling processor will randomly sample the TDM data and use the samples to compute the adaptive nulling weights. (A correlation matrix is formed, and the Householder algorithm is used to reduce this matrix to its lower triangular form. The weights are then calculated by back substitution [1].) The adaptive nulling processor must have access to all of the data samples output by the filters. Therefore, downsampling the data prior to broadcast on the TDM bus is not possible. Thus, the eight 1.5-MHz complex data streams are combined into one 12-MHz complex data stream. Each front end module is assigned a 42-ns interval during which it can drive its data on to the bus. (This interval allows for a 42-ns buffer between each time slot.)

Each front end module contains the filters for one receiver channel and the circuitry for one half beamformer. Therefore, each module must contain both the transmit and receive logic required to implement the TDM bus. The transmit and receive control logic can be combined to simplify the design and reduce the need for intermodule timing signals.

2.3.1 TDM Bus Hardware

Each data sample consists of a 16-bit inphase component and a 16-bit quadrature component. These two words are broadcast simultaneously along parallel inphase and quadrature buses. Because a single point failure on either of these buses could result in total failure of the signal processor, the front end subsystem provides a fully redundant copy of each bus. Thus, the composite TDM bus consists of primary inphase and quadrature buses and spare inphase and quadrature buses. Each bus is 16 bits wide. Physically, the TDM bus claims the entire J3 backplane of the front end subsystem rack. This backplane is populated with 96 pin DIN connectors. The pinout of these connectors is presented in Table 3. The signal names ITDMA and QTDMA refer to the primary inphase and quadrature TDM buses. Likewise, the signal names ITDMB and QTDMB refer to the spare inphase and quadrature TDM buses.

TABLE 3

TDM Bus Backplane Pinout

	P3A	P3B	P3C		P3A	P3B	P3C
Pin 1	QTDMA[0]	GND	QTDMB[0]	Pin 17	ITDMA[0]	GND	ITDMB[0]
Pin 2	QTDMA[1]	GND	QTDMB[1]	Pin 18	ITDMA[1]	GND	ITDMB[1]
Pin 3	QTDMA[2]	GND	QTDMB[2]	Pin 19	ITDMA[2]	GND	ITDMB[2]
Pin 4	QTDMA[3]	GND	QTDMB[3]	Pin 20	ITDMA[3]	GND	ITDMB[3]
Pin 5	QTDMA[4]	GND	QTDMB[4]	Pin 21	ITDMA[4]	GND	ITDMB[4]
Pin 6	QTDMA[5]	GND	QTDMB[5]	Pin 22	ITDMA[5]	GND	ITDMB[5]
Pin 7	QTDMA[6]	GND	QTDMB[6]	Pin 23	ITDMA[6]	GND	ITDMB[6]
Pin 8	QTDMA[7]	GND	QTDMB[7]	Pin 24	ITDMA[7]	GND	ITDMB[7]
Pin 9	QTDMA[8]	GND	QTDMB[8]	Pin 25	ITDMA[8]	GND	ITDMB[8]
Pin 10	QTDMA[9]	GND	QTDMB[9]	Pin 26	ITDMA[9]	GND	ITDMB[9]
Pin 11	QTDMA[10]	GND	QTDMB[10]	Pin 27	ITDMA[10]	GND	ITDMB[10]
Pin 12	QTDMA[11]	GND	QTDMB[11]	Pin 28	ITDMA[11]	GND	ITDMB[11]
Pin 13	QTDMA[12]	GND	QTDMB[12]	Pin 29	ITDMA[12]	GND	ITDMB[12]
Pin 14	QTDMA[13]	GND	QTDMB[13]	Pin 30	ITDMA[13]	GND	ITDMB[13]
Pin 15	QTDMA[14]	GND	QTDMB[14]	Pin 31	ITDMA[14]	GND	ITDMB[14]
Pin 16	QTDMA[15]	GND	QTDMB[15]	Pin 32	ITDMA[15]	GND	ITDMB[15]

The front end module uses 74ACT374 octal D flip-flops with tristate outputs to both drive data onto and receive data from the TDM bus. The implementation of the redundant bus structure at the transmit end consists simply of a second set of 74ACT374s with identical input connections. Thus, the data and timing on the primary and spare TDM buses are identical. At the receive end, data is captured from both the primary and spare buses with the 74ACT374 octal D flip-flops. The outputs of these devices are wired together, bit by bit, and only the devices corresponding to the selected bus have their tristate outputs enabled. The selection of the primary (Bus A) or the redundant (Bus B) TDM buses is made by programming a bit in the front end module's Discrete Control Register (DCR) via the VMEbus interface. The specific address and bit assignment of this memory mapped register will be described in Section 2.6.3. The selection of buses is arbitrary in a fully functional ground-based system, but a deployed system will require that the adaptive nulling processor execute TDM bus diagnostics to determine which bus (if either) is functional and program the front end modules accordingly. Figure 14 is a block diagram of the transmit and receive portions of the redundant inphase and quadrature buses. The signals CH1N, CH2N, ..., CH8N (denoted as CHxN in the figure) are used by each of the eight modules to enable their TDM bus drivers during the appropriate time slot. The figure also depicts a single signal used to enable/disable the primary/spare buses with one set of receivers enabled "active low" and the other enabled "active high." This is an oversimplification of the actual circuitry but is logically correct with respect to the TDM bus.

2.3.2 TDM Bus Timing

The 24-MHz System Clock. The TDM bus uses the 24-MHz system clock to derive its timing and control signals. Correct operation of the bus requires that the 24-MHz clock, received by each of the eight modules, has minimal skew. The system clock board, residing in the center of the subsystem rack, uses AT&T 41MP differential drivers to drive two versions of the 24-MHz clock down to each end of the twisted-pair wirewrapped backplane. These signals are terminated at the source with 220-ohm resistors and on the ends of the backplane with 110-ohm resistors. The clock signal is received on each front end module with the AT&T 41MF differential receiver. The TTL output of this receiver is then connected to the inputs of four 74LS240 buffer inverters. The four outputs of these buffers (the signals 24MHZA, 24MHZB, 24MHZC, and 24MHZD) are distributed throughout the front end module. Attempts have been made to keep the capacitive loading on each of these four lines as even as possible. Because the same version of the clock is used on each module to generate the TDM bus timing, these signals essentially have identical capacitive loads. Figure 15 shows the clock distribution and buffering scheme. The total clock skew between each of the eight front end modules is the sum of the variations between the differential receivers and the 74LS240 buffers. The AT&T 41MF receivers contribute a relatively low delay (2.0 ns) with skew variations of only a fraction of that. However, the 74LS240 creates an uncertainty of up to 10 ns. This number is tempered by the fact that the eight modules will be running at the same temperature and driving the same load. The 74LS240 is preferred over faster versions of this part (such as the 74ACT240 or the 74FCT240) because the slower rise and fall times result in less ringing on the wire wrap boards.

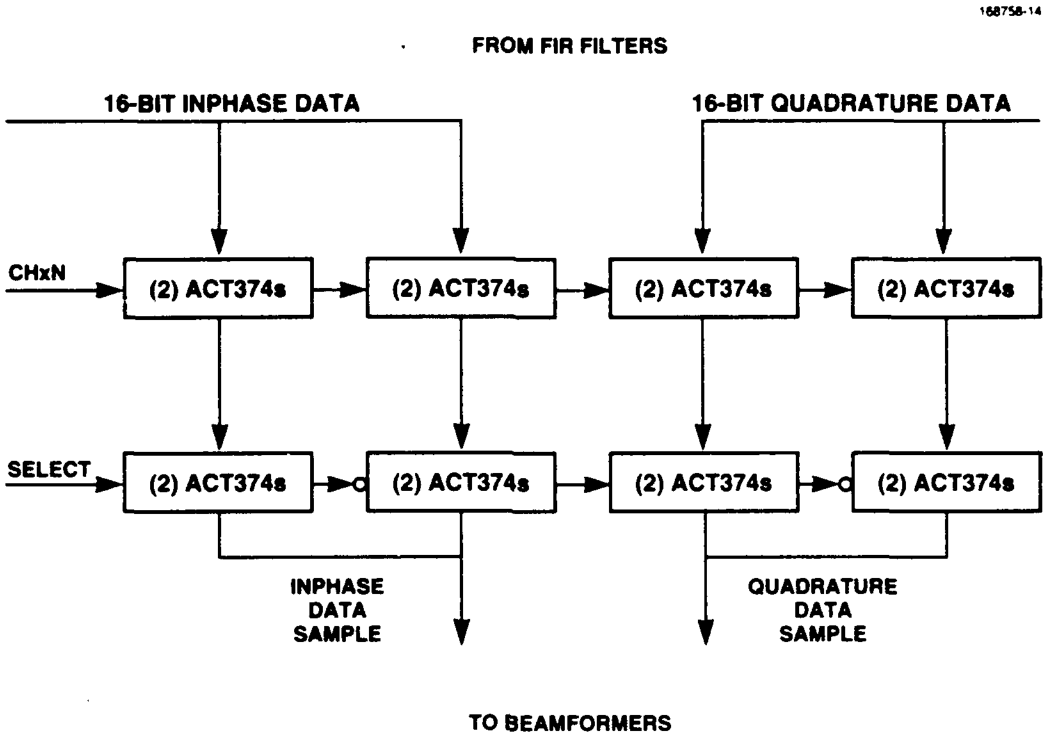


Figure 14. Block diagram of primary/spare TDM bus architecture.

TDM Bus Timing and Control. The data samples from each module become available at the output of the FIR filters at the same edge of the 24-MHz system clock. When this edge occurs, a TDM bus cycle is initiated in which the eight time slots are allocated and used by the eight front end modules to broadcast their respective data samples. The TDM bus timing logic is not continually running; rather, it operates in a start/stop mode with a cycle occurring whenever a data sample becomes available. When data is input to the front end module at the maximum allowable rate of 4.5 MHz, the time slot allocation on the TDM bus becomes continuous with the first time slot of a sample immediately following the eighth time slot of the previous sample. This timing is the normal mode of operation.

The alternating inphase and quadrature sample components are clocked from the FIR filter chips with the signal OUTRDY. (This signal is generated by the front end module shift register and not the A100s.) The timing diagram in Figure 16 depicts a TDM bus cycle being initiated by the signal TDMLD one clock cycle after the arrival of an inphase data sample. The quadrature portion of this sample arrives in the middle of the bus cycle, is latched with the signal QLOAD, but is not broadcast until the next cycle. The front end modules operate in lockstep synchrony, triggered by the simultaneous input of the radar A/D samples. A predetermined identical number

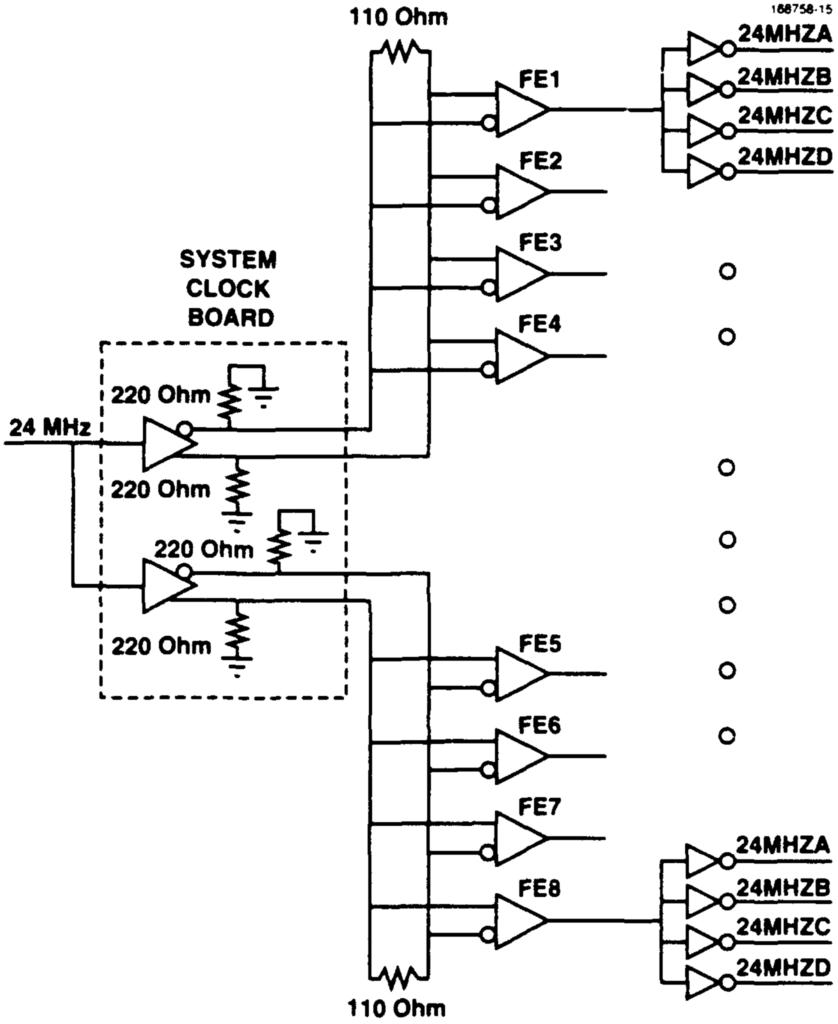


Figure 15. Front end subsystem clock distribution and buffering.

of 24-MHz clock cycles after the data sample is input, each module is ready with a complex output data sample. Thus, each module can independently generate its own TDM bus timing signals and rely on each of the other modules to be fully synchronized. Each module determines which time slot it is assigned by reading the three Module ID bits tied high or low on the backplane. (The front end modules are fully identical and perform all channel-specific functions according to these Module ID bits.) With its channel identification, each module generates its version of the signal CHxN (where x is the module number 1 through 8) to enable its TDM bus drivers. The signals CH1N through CH8N are also driven on to the backplane. The adaptive nulling processor can use these signals, along with the system clock, to capture and interpret data from the TDM bus. Each signal is transmitted individually to allow for module sparing. If one or more front end modules fail, they can simply be turned off by the adaptive nulling processor, and the remaining data strobes and their corresponding time slots will remain valid.

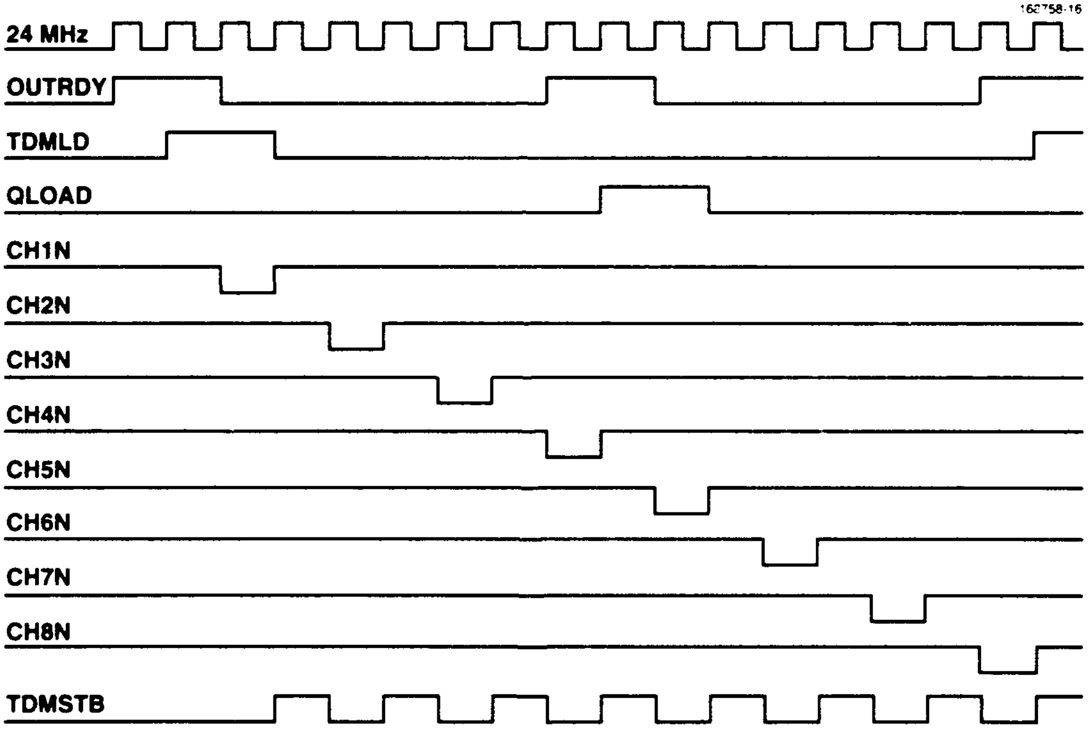


Figure 16. TDM bus timing diagram.

Because each of the front end modules also contains half of a beamformer, the modules must also collect all eight samples driven onto the bus. The same logic used to generate the CHxN signal also generates an eight-clock sequence called TDMSTB, which is also depicted in Figure 16. The rising edges of this signal are used to clock the data off the TDM bus. Note that all eight time slots are always captured with this signal. The beamformer computes a weighted sum of the eight samples. These weights are programmed by the adaptive nulling processor. If one or more front end modules have been identified as failing, the weights for these modules can simply be assigned to zero. Provided that the failure mode is not destructive to the TDM bus, this approach logically eliminates the failed module from the system. The failure of a front end module does, however, result in limiting the signal processor test bed to a subset of its algorithms [1].

Due to the high-speed operation of this bus, the logic used to generate the TDM bus timing signals is critical. The recommended timing to be used by the adaptive nulling processor in reading data from the TDM bus is presented in Figure 17. The figure shows the 24-MHz system clock and its timing relationship to the CHxN group of signals. The clock signal can be ORed with each of the signals (CH1N through CH8N) to create eight independent data strobes. The rising edge of the ensuing data strobes can then be used to latch the data onto the adaptive nulling processor board. The eight CHxN signals are used to enable the tristate outputs of the TDM bus drivers on each of the front end modules. The rising edge of the data strobe is designed to be coincident with the end of this enable pulse. This timing maximizes the setup time and minimizes the hold time at the input to the receiving logic. This scheme closely matches the timing used by the beamformer portion of the front end modules and preserves the fault tolerance that channel sparing of the eight modules provides.

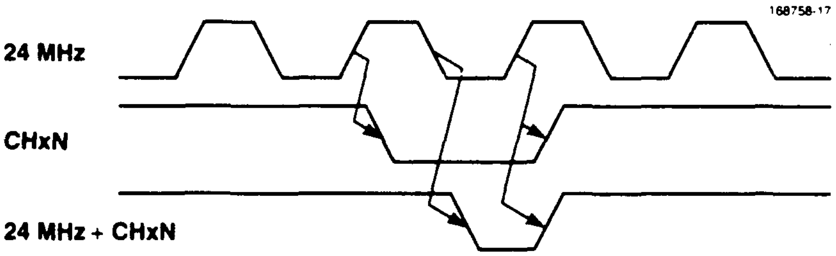


Figure 17. TDM bus-to-adaptive nulling processor interface timing.

TDM Bus Data Interleaving. The corresponding inphase and quadrature portions of each data sample are clocked out of the FIR filter cascade on an alternating basis. The timing diagram in Figure 16 shows that the TDM bus cycle is initiated immediately after the arrival of the inphase sample and that the corresponding quadrature sample arrives halfway through the TDM bus cycle. Because the module assigned to the first time slot must have both the inphase and quadrature components available as soon as the bus cycle is initiated and all eight modules are interchangeable, all eight modules must have the data available at the start of the cycle. Thus, for each TDM bus cycle, the quadrature sample that arrives in the middle of the cycle is latched and broadcast along with the inphase component of the next cycle. That is, each TDM bus cycle broadcasts the Nth inphase sample and the (N-1)st quadrature sample. Figure 18 graphically demonstrates this shifting of the quadrature data.

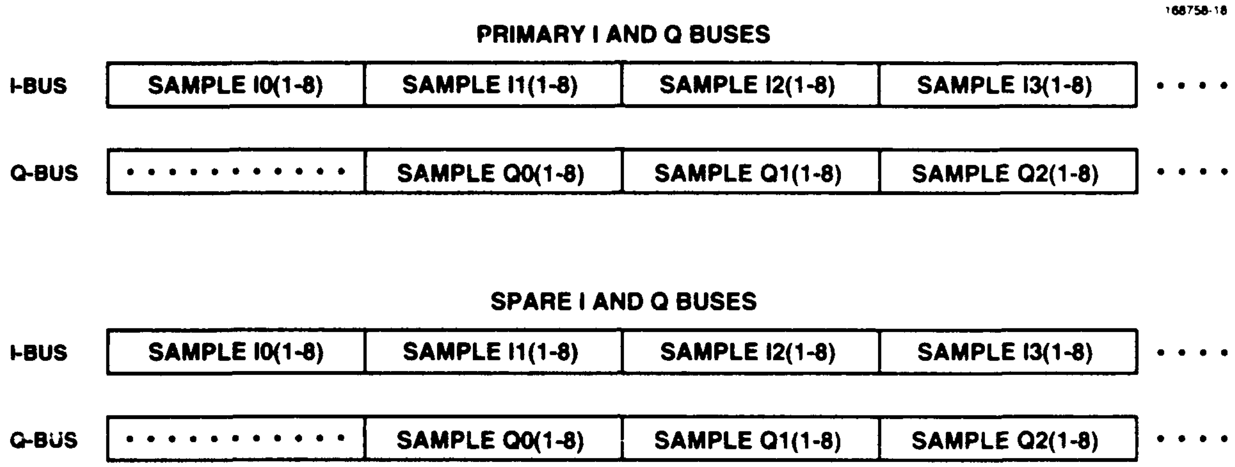


Figure 18. Offset inphase and quadrature data transmission.

This offset of inphase and quadrature data is very convenient for the beamformer portion of the front end module. The adaptive nulling processor requires access to all of the data sample's output by the filters in generating the beamformer weights. However, the beamformers calculate the weighted sum of only every other sample broadcast on the TDM bus. Thus, the beamformer logic can be simplified by receiving data from the inphase and quadrature buses on alternate cycles. This approach results in the desired downsampling while maintaining a constant data rate at the beamformer input. This scheme does, however, create a data offset problem for the adaptive nulling processor, which can be solved by using offset address generators at the TDM bus interface or simply by using offset addressing in the software written for the adaptive nulling processor.

2.4 Beamformers

Adaptive nulling or interference suppression in the signal processor test bed is accomplished by side lobe canceling. The basic approach is to use multiple auxiliary antenna channels to cancel the interference. The adaptive nulling processor employs various algorithms [1] to process the TDM bus data. The output of these algorithms is a set of weights that, when applied to the eight channels of data, simultaneously form main channels while using the auxiliaries to null interference. These weights are downloaded to the front end modules and applied by the beamformers. To be effective, the weights must be recomputed and updated every 10 ms [1]. In radar systems the function block that we call *beamformer* is also known as *combiner* or *digital combiner*. The front end subsystem includes four beamformers. Depending on the operation mode of the radar, up to four main beams may be required. (The various operation modes are discussed by Pohlig [1].)

2.4.1 Beamformer Architecture

The beamformer's task is to compute a weighted sum of the eight channels of radar data, which means that each complex data sample must be multiplied by a complex weight and the eight products must be summed. A complex multiply operation, $x * y$, is of the form

$$(\mathrm{Re}[x] + j\ \mathrm{Im}[x]) * (\mathrm{Re}[y] + j\ \mathrm{Im}[y]) = (\mathrm{Re}[x]*\mathrm{Re}[y] - \mathrm{Im}[x]*\mathrm{Im}[y]) + j\ (\mathrm{Re}[x]*\mathrm{Im}[y] + \mathrm{Im}[x]*\mathrm{Re}[y])$$

and requires four multiply operations and two additions. Thus, one time sample of TDM data requires 32 multiplications and 32 additions. This calculation is accomplished in the 16 12-MHz clock cycles (1.33 μs) afforded by the 2X downsampling of the TDM bus.

The beamformer employs two multiply accumulators (MACs), one for the real part of the output and one for the imaginary part. Each multiply accumulator must receive both the real and imaginary parts of the eight data samples. Figure 19 presents the beamformer architecture and the appropriate data streams as they enter the MACs.

In addition to connecting the outputs of the primary and redundant TDM buses, the front end module also connects the inphase (real) and quadrature (imaginary) buses together. The control logic is required to select between the primary and redundant buses, as described in Section 2.3. Additionally, the control logic alternates between the real and imaginary bus data. Because the inphase and quadrature TDM bus data is offset (see Figure 18), this alternating pattern yields both the real and imaginary half of every other sample on the TDM bus arriving, as shown in Figure 19. This figure uses the notation R1,....R8 to designate the real half of the data samples as well as the real half of the complex weights. Likewise, it uses I1,....I8 to designate the imaginary (or quadrature) half of the data stream and complex weights. The real and imaginary portions of the weights are stored redundantly in the beamformer memories. Cycling through successive addresses yields the coefficient data stream depicted in Figure 19. The MACs multiply the two input data streams and maintain a running sum. After each 16-word cycle the sum is latched out of the MAC, and the internal accumulator is cleared for the next cycle.

The dotted line in Figure 19 encircles one half of a beamformer. This half beamformer includes a MAC, a dual-port RAM, a four-bit counter, and the miscellaneous logic associated with the VMEbus interface and the timing and control. It is easy to see how the four beamformers are conveniently subdivided into eight half beamformers and distributed between the eight front end modules. The real and imaginary halves of the beamformer are identical. The nature of the

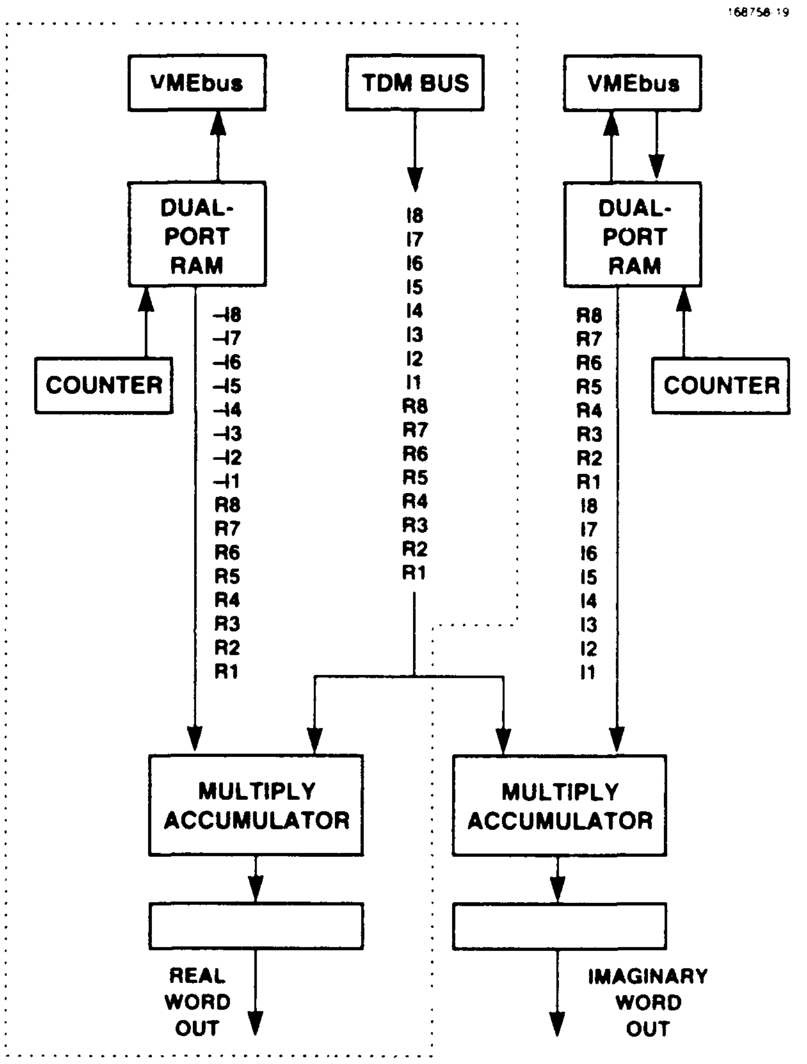


Figure 19. Architecture of a full beamformer.

data they produce is strictly dependent on the arrangement of the coefficients in the RAM. As will be described in Section 2.5, the outputs of the corresponding real and imaginary beamformer halves are wired together and time-division-multiplexed in a similar fashion to the TDM bus. This arrangement necessitates the assignment of beamformer halves on the eight modules as shown in Table 4. The Module ID bits, read from the backplane, instruct the module to its assigned beamformer half.

TABLE 4

Front End Module Beamformer Assignment

Front End Module	Beamformer Assignment
1	BF 1 - Real
2	BF 1 - Imaginary
3	BF 2 - Real
4	BF 2 - Imaginary
5	BF 3 - Real
6	BF 3 - Imaginary
7	BF 4 - Real
8	BF 4 - Imaginary

2.4.2 Beamformer Weights — Storage and Updating

The beamformers employ a dual-port RAM (two IDT 7132s) to store the beamformer weights. One RAM port is configured for reading and writing and is memory mapped into the VMEbus address space. (The VMEbus interface and addressing will be discussed in Section 2.6.1.) The other port is configured as a read-only port and is addressed by the beamformer address counter and control logic. Each IDT 7132 is an 8-bit by 2048-bit device. The two ICs are connected in parallel to create a 16-bit-wide memory. The memory map for the 2-kbit array is shown in Figure 20.

Two blocks of 16-bit coefficients are allocated. The beamformer address counter cycles address lines ADR0 to ADR3 through the 16 coefficients. The alternate blocks of coefficients are selected by changing the state (0 or 1) of the most significant address bit of the RAMs. This bit is connected directly to the Discrete Control Register (DCR) of the front end module. This register, described in Section 2.6.3, is VMEbus memory mapped and loaded by the adaptive nulling processor. It has been noted [1] that updates to the beamformer weights must occur simultaneously across all eight

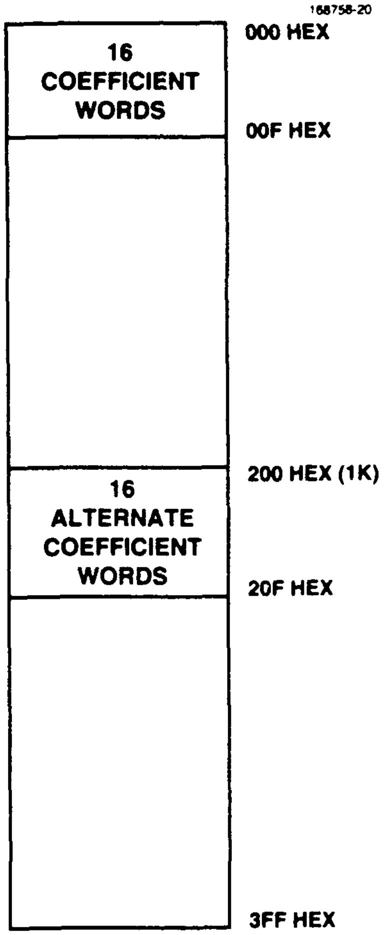


Figure 20. Dual-port RAM memory allocation.

channels. Therefore, all beamformer weights must be changed while the front end modules are inactive. To expedite this, the adaptive nulling processor can load the updated set of coefficients into the currently unused bank of beamformer coefficient memory and then simply swap banks on all eight modules by writing to the respective DCRs. This bank swap should still be performed while the module is inactive (for example, during the transmit interval).

The beamformer port on the dual-port RAM is permanently enabled. Regardless of the state of the counter, some address is always being accessed. A VMEbus write operation to this address will not be successful. Therefore, the software must keep track of which coefficient bank is currently selected and load coefficients to the opposite bank. (In the normal mode of operation, the software will be required to maintain the dual-port RAM selection status regardless of this port conflict issue.)

2.5 Beamformer Output Bus

The weighted sum of the eight channels of data constitutes a formed radar beam that has been frequency shifted to baseband, demodulated into inphase and quadrature components, and pulse compressed (match filtered with the transmit pulse). The beam's jammer interference has been adaptively nulled. The data is now ready for Doppler processing, clutter cancellation, and target detection. These tasks are all performed by the vector processor subsystem. Each of the four output data streams from the four beamformers is transmitted to the vector processors via the Beamformer Output bus.

2.5.1 Data Transfer Timing

The Beamformer Output bus consists of four beamformer channels, operating in parallel, each transferring 16-bit inphase and 16-bit quadrature data at a rate of 750,000 complex samples per second. Each beamformer uses an 8-bit-wide data path and time-division-multiplexes the I and Q components of the data sample, as well as the high and low bytes of each component. The byte-wide output of two adjacent front end modules are wired together and, as with the TDM bus, the synchronous modules enable their output drivers only during the appropriate time slots. Three control lines accompany the byte-wide output. These lines are generated by both halves of the beamformer. Because they are identical, only one set of these control lines need be sent to the vector processors.

Figure 21 shows the signal and timing requirements associated with the transfer of one data sample on the Beamformer Output bus. In addition to the eight data bits, BFDATA (0-7), the diagram shows the three control lines. The signal REAL/IMAGN indicates whether the data byte belongs to the real or imaginary half of the data sample. The signal HBYTE/LBYTEN tags the data byte as the high or low half of the 16-bit word. A third signal DATSTB is to be used by the vector processors to latch the data byte. The rising edge of this signal occurs at the center of the interval during which the data output byte is enabled.

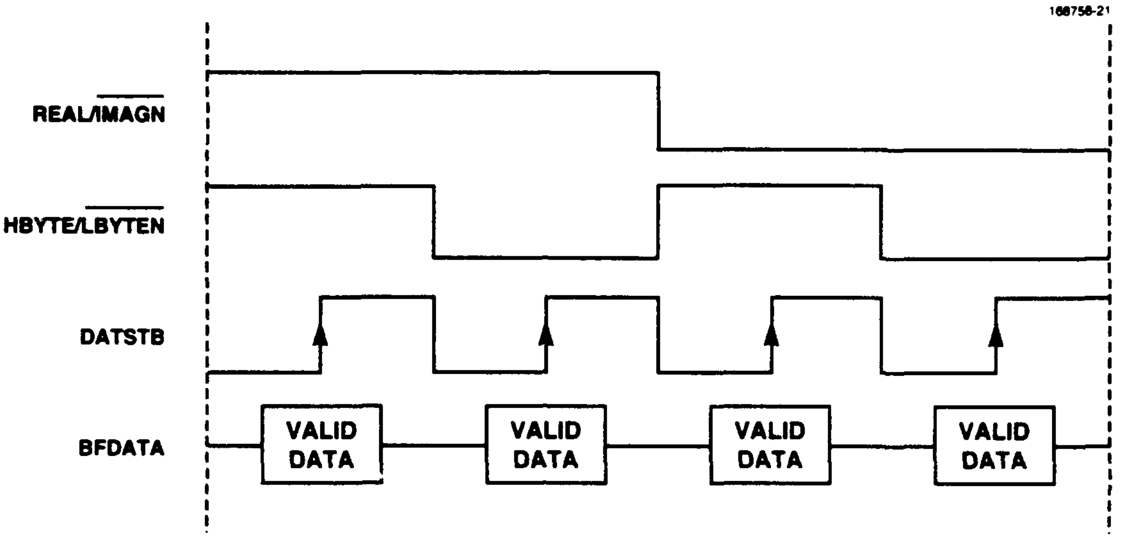


Figure 21. Beamformer Output bus timing.

The data rate of the bus and the frequency of DATSTB will be 3 MHz as long as the front end filters are operating on a steady stream of incoming data at a 4.5-MHz rate. All aspects of the front end module operate in a start/stop mode. Each set of six input samples from the radar receiver result, after some known time delay, in one timing window, as depicted in the figure. If the 4.5-MHz input rate is sustained, then the consecutive timing windows will be adjacent. However, an input data rate that is even slightly slower than 4.5 MHz will result in an occasional "extra" clock cycle inserted between the output epochs. And by extension, if the data rate is much slower than 4.5 MHz, then some number of 24-MHz clock cycles will fill the gap between each interval. The state of the control and data lines during the gap are unimportant and will not be specified except that there will not be a rising edge on the DATSTB signal. The vector processor must simply capture data on the rising edge of DATSTB and interpret this data according to the state of the signals REAL/IMAGN and HBYTE/LBYTEN at the time of this rising edge.

2.5.2 Hardware Interconnect

The four 11-signal subbuses that compose the Beamformer Output bus (along with matching twisted-pair grounds) are available on the wirewrap P2 backplane of the subsystem card cage. The physical interconnect is accomplished by the following scheme. The data buses of the matching beamformer halves are combined with twisted-pair wires. Each of the four byte-wide buses, along with its control signals, are then wired with twisted pairs along the backplane over to a single unused connector. This is a 96-pin DIN connector located in the P2 position at backplane slot number 11 (i.e., centered on the backplane). An identical wiring scheme is used in the vector processor subsystem. The two subsystems can then be connected by a 96-wire cable that slips over the backplane wirewrap posts on each of the two racks. Tables 5 and 6 give the pinout of the dedicated connector on the front end subsystem backplane.

TABLE 5

Pin Assignment for Beamformer Output Bus Inter-Rack Cabling (Beamformers Numbers 1 and 2)

Signal Name	Signal Pin	Ground Pin
BF1DAT⟨7⟩	J10P2A01	J10P2A02
BF1DAT⟨6⟩	J10P2A03	J10P2A04
BF1DAT⟨5⟩	J10P2A05	J10P2A06
BF1DAT⟨4⟩	J10P2A07	J10P2A08
BF1DAT⟨3⟩	J10P2A09	J10P2A10
BF1DAT⟨2⟩	J10P2A11	J10P2A12
BF1DAT⟨1⟩	J10P2A13	J10P2A14
BF1DAT⟨0⟩	J10P2A15	J10P2A16
DATSTB1	J10P2B03	J10P2B04
HB/LBN1	J10P2B05	J10P2B06
REAL/IMAGN1	J10P2B07	J10P2B08
BF2DAT⟨7⟩	J10P2A17	J10P2A18
BF2DAT⟨6⟩	J10P2A19	J10P2A20
BF2DAT⟨5⟩	J10P2A21	J10P2A22
BF2DAT⟨4⟩	J10P2A23	J10P2A24
BF2DAT⟨3⟩	J10P2A25	J10P2A26
BF2DAT⟨2⟩	J10P2A27	J10P2A28
BF2DAT⟨1⟩	J10P2A29	J10P2A30
BF2DAT⟨0⟩	J10P2A31	J10P2A32
DATSTB2	J10P2B09	J10P2B10
HB/LBN2	J10P2B11	J10P2B14
REAL/IMAGN2	J10P2B15	J10P2B16

TABLE 6

Pin Assignment for Beamformer Output Bus Inter-Rack Cabling (Beamformers Numbers 3 and 4)

Signal Name	Signal Pin	Ground Pin
BF3DAT⟨7⟩	J10P2C01	J10P2C02
BF3DAT⟨6⟩	J10P2C03	J10P2C04
BF3DAT⟨5⟩	J10P2C05	J10P2C06
BF3DAT⟨4⟩	J10P2C07	J10P2C08
BF3DAT⟨3⟩	J10P2C09	J10P2C10
BF3DAT⟨2⟩	J10P2C11	J10P2C12
BF3DAT⟨1⟩	J10P2C13	J10P2C14
BF3DAT⟨0⟩	J10P2C15	J10P2C16
DATSTB3	J10P2B17	J10P2B18
HB/LBN3	J10P2B19	J10P2B20
REAL/IMAGN3	J10P2B23	J10P2B24
BF4DAT⟨7⟩	J10P2C17	J10P2C18
BF4DAT⟨6⟩	J10P2C19	J10P2C20
BF4DAT⟨5⟩	J10P2C21	J10P2C22
BF4DAT⟨4⟩	J10P2C23	J10P2C24
BF4DAT⟨3⟩	J10P2C25	J10P2C26
BF4DAT⟨2⟩	J10P2C27	J10P2C28
BF4DAT⟨1⟩	J10P2C29	J10P2C30
BF4DAT⟨0⟩	J10P2C31	J10P2C32
DATSTB4	J10P2B25	J10P2B26
HB/LBN4	J10P2B27	J10P2B28
REAL/IMAGN4	J10P2B29	J10P2B30

The pin assignment for the control signals appears arbitrary and somewhat confusing. Because the front end subsystem is VMEbus compatible,[4] some of the pins in the B row of the P2 connector are reserved for power and ground. Although this connector is not used for VMEbus purposes, avoiding the use of these power and ground pins seemed prudent.

2.6 Miscellaneous Front End Module Functions

2.6.1 The VME Bus Interface

For all of its operations, the front end subsystem requires that coefficients and control words be downloaded by a host processor. In the signal processor test bed, this task is performed by the adaptive nulling processor. To the host processor memory interface, the front end appears like a large memory array with the caveats that the array is not contiguous and some of the locations are write-only (i.e., read operations to these addresses are undefined and cannot be performed). The front end module memory interface is a simple, bare-bones asynchronous implementation. For convenience, the interface has been designed to be compatible with a subset of the VMEbus specification [2]. Rather than try to describe the numerous aspects of the VMEbus that are not supported, this section will simply define the front end memory interface in the context of the VMEbus.

The front end modules act as VMEbus slaves and are limited to the 24-bit address space and the 16-bit data bus that are available on the J1 connector of the VMEbus. (The J2 connector is not used for VMEbus at all.) The VMEbus address modifier bits are decoded to determine whether the standard (24-bit) address space is being addressed. (It decodes for 39 Hex, Standard Non-Privileged Data Access.) The front end supports 16-bit data transfers only. Attempts to perform byte or longword read or write operations will result in a bus error. The IACK signal is decoded to ensure that the bus cycle is not an interrupt cycle. And finally, the VMEbus data strobes (DS0*. DS1*). address strobe (AS*), data acknowledge signal (DTACK*), and read/write signal (WRITE*) are used to generate the internal control signals. The power pins on the J1 and J2 connectors are also VMEbus compatible.

The front end module uses five programmable logic devices along with the appropriate buffers and registers to implement the interface. The read and write cycle delays are fewer than optimal but suffice for the application. The module allows two 24-MHz clock cycles to decode the VMEbus address and two more cycles to capture or output the data.

Front End Subsystem Address Decode. Each of the eight front end modules has eight A100s, two 16-word beamformer coefficient banks and a Discrete Control Register. These devices are mapped into the 24-bit VMEbus address, as shown in Figure 22.

[4] Actually, only a small subset of the VMEbus specification is implemented. See Section 2.6.1.

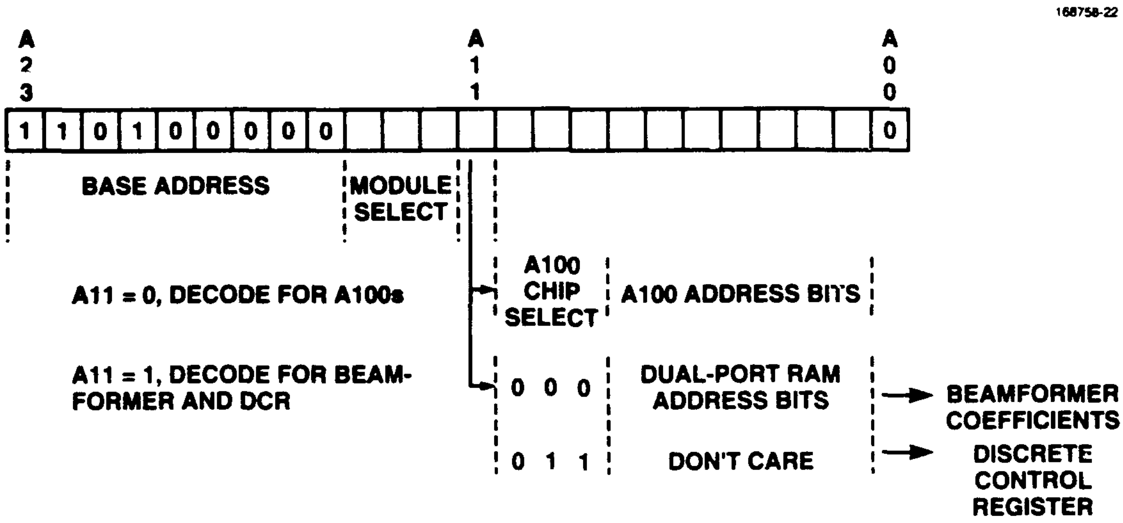


Figure 22. Front end subsystem address decoding.

In the VMEbus specification, address bit A00 is used to select individual bytes. Because the front end supports full 16-bit transfers only, this bit is always zero. The base address, as shown in the figure, can be modified by reprogramming the programmable logic devices on all eight modules.

Three pins on the J2 connector are designated as the Module Identification bits. These bits are wirewrapped to Vcc and ground on the backplane according to Table 7. The Module ID bits are compared with the Module Select field in the VMEbus address as part of the address decode cycle. If the two bits match, then a data acknowledge cycle is initiated, and the address word is decoded further.

Address bit A11 determines whether the VMEbus transaction will be directed to the A100s (A11 = 0) or to the beamformer coefficients or DCR (A11 = 1). The remaining address bits are decoded according to the state of this bit. When address bit A11 selects the A100s, address bits A10, A09, and A08 are decoded to select the eight A100s according to Table 8. The A100s designated I/Q1, I/Q2, and I/Q3 perform the inphase and quadrature filtering. The numbers correspond to the assignment of successive time samples to each A100 in the structure. The A100s designated EQPC1 to EQPC5 are the cascade of five A100s that perform the convolved functions of channel equalization and pulse compression. The numbering relates to the spatial arrangement of the devices on the board and on the schematics. However, because of the transversal filter architecture, EQPC5 actually is loaded with the low-order coefficients and EQPC1 with the high-order coefficients. The remaining address bits are then decoded internally by the A100. The internal memory map of the A100 is defined by the Inmos literature [4]. Section 3, on programming the front end modules, will present this map in the context of the front end filters.

TABLE 7

Front End Module Identification Bits

Card Cage Slot	Module Number	ID2	ID1	ID0
1	1	0	0	0
2	2	0	0	1
3	3	0	1	0
4	4	0	1	1
7	5	1	0	0
8	6	1	0	1
9	7	1	1	0
10	8	1	1	1

TABLE 8

Decoding the A100 Select Field

A10	A09	A08	A100 Designation
0	0	0	I/Q1
0	0	1	I/Q2
0	1	0	I/Q3
0	1	1	EQPC1
1	0	0	EQPC2
1	0	1	EQPC3
1	1	0	EQPC4
1	1	1	EQPC5

When address bit A11 is one and address bit A10 is zero, the front end module does not decode the remaining address. This portion of the memory space is reserved for use by the system clock board and the adaptive nulling processor.

When address bit A11 is one and address bit A10 is one, address bits A09 and A08 are used to select between the DCR and the beamformer dual-port RAM. If A09 and A08 are both one, then the DCR is selected. If A09 and A08 are both zero, then the dual-port RAM is selected. In the latter case, all seven remaining address bits are connected to the RAM. While the host processor can read or write to this entire memory space, only addresses 0 through 16 are defined. (The alternate bank of beamformer coefficients is selected by toggling a bit in the DCR.)

Attempts to read or write from any address not specified above will not be acknowledged by the front end in any manner.

Table 9 presents the memory map of one front end module. The starting addresses are given for each of the eight modules.

2.6.2 Data Input Zeroing

Digital FIR filters, much like analog filters, operate on a continuous stream of data. The transient response introduced into the output data stream at startup will depend on the initial state of the filter. Similarly, if we choose to pad the end of the data stream in order to pump all of the data out of the filter, the choice of this data will introduce artifacts in the filter output. The signal processor operates on finite data streams composed of the data sampled between transmit pulses of the radar. Because of transient effects, the filtered data at the beginning and end of this finite data stream (up to the length of the filter) may be of little or no value to subsequent signal-processing algorithms. However, it is convenient for test and analysis purposes to know exactly what this data will be. It was therefore decided that the front end subsystem would support a test mode in which the input data stream could be forced to all zeros. This mode could be used both to clear the filters initially or to pad the end of the data stream with zeros.

The approach requires that the adaptive nulling processor program each front end module with the number of valid data samples it is to accept in each pulse repetition interval (PRI). The PRI pulse (used to demarcate PRIs) is used to clear a set of counters. These counters are then incremented each time a new data sample is received from the radar receiver. When the count becomes equal to the previously programmed limit, the front end module asserts the asynchronous clear signal on the input data buffers. (See Section 2.2.1 for a description of these buffers.) Any additional Datardy pulses after this point result in the insertion of zeros into the FIR filter cascade. The timing for this process is shown in Figure 6. This technique requires that the radar receiver (or Test Vector Generator) transmit some number of "extra" Datardy pulses each PRI. To clear the filters completely, this number can be up to 288 pulses. (It may be sufficient to partially clear the filters.) This operating mode is not an anticipated mode of the signal processor when processing actual radar returns, but it is a desirable mode for test purposes.

TABLE 9

Front End Module Memory Map

Module	Hex Starting Address by Module Number							
Memory Map	1	2	3	4	5	6	7	8
A100 - I/Q1	D00000	D01000	D02000	D03000	D04000	D05000	D06000	D07000
A100 - I/Q2	D00100	D01100	D02100	D03100	D04100	D05100	D06100	D07100
A100 - I/Q3	D00200	D01200	D02200	D03200	D04200	D05200	D06200	D07200
A100 - EQPC1	D00300	D01300	D02300	D03300	D04300	D05300	D06300	D07300
A100 - EQPC2	D00400	D01400	D02400	D03400	D04400	D05400	D06400	D07400
A100 - EQPC3	D00500	D01500	D02500	D03500	D04500	D05500	D06500	D07500
A100 - EQPC4	D00600	D01600	D02600	D03600	D04600	D05600	D06600	D07600
A100 - EQPC5	D00700	D01700	D02700	D03700	D04700	D05700	D06700	D07700
Beamformer Coefficient DP-RAM	D00800	D01800	D02800	D03800	D04800	D05800	D06800	D07800
Not Decoded	D00900	D01900	D02900	D03900	D04900	D05900	D06900	D07900
Loosely Decoded For DCR	D00B00	D01B00	D02B00	D03B00	D04B00	D05B00	D06B00	D07B00
Not Decoded	D00C00	D01C00	D02C00	D03C00	D04C00	D05C00	D06C00	D07C00

2.6.3 The Discrete Control Register

Each front end module contains one 16-bit Discrete Control Register (DCR), which is constructed from two 74ACT377 octal D flip-flops with clock enable. This register is memory mapped into the VMEbus address space, as described in Section 2.6.1. The host processor can write to this register, but the contents cannot be read back.

TABLE 10

Format of the Discrete Control Register

Bit Number	Signal
0	LSB
1	
2	
3	Number
4	of
5	valid
6	samples
7	per
8	PRI
9	
10	
11	MSB
12	Unused
13	COEFSEL
14	TDMSELB/AN
15	CHDISABL

The 16-bit register is formatted as shown in Table 10. The lower 12 bits are programmed with the number of valid data samples expected in each PRI. As described in Section 2.6.2, this number is fed into a comparator along with a Datardy pulse counter. Note that 12 bits (allowing for 4096 valid samples) are more than sufficient for the anticipated operation of the system.

The signal COEFSEL selects between the two banks of beamformer coefficients. As described in Section 2.4, use of this bit is mandatory in programming the beamformer coefficients.

The signal TDMSELB/AN selects between the primary and redundant TDM buses. Provided the sy^ste^rr has not encountered any TDM bus failures, the state of this bit will be transparent to the user. When this signal is zero, the primary bus (Bus A) is selected, and when the signal is one, the alternate bus (Bus B) is enabled.

If a failure mode occurs, resulting in a single front end module broadcasting data in every TDM time slot, then a single point failure of the system would result. In anticipation of this situation, the signal CHDISABL allows the adaptive nulling processor to effectively disable a given front end module, thus circumventing the failure. Setting bit 15 of the DCR to one. on any module, will prohibit that module from broadcasting data on the TDM bus. Because of the inherent capacitance of the bus, if a module is turned off and does not transmit data during its TDM bus time slot, the beamformers and adaptive nulling processor will generally observe data from the previous time slot. The adaptive nulling processor must compute the nulling weights to zero out this data.

3. PROGRAMMING THE FRONT END SUBSYSTEM

The front end subsystem is programmable in that changing the contents of various memory locations or registers, the transfer function of the system can change. It is not programmable in the same sense as a microprocessor or a general-purpose computer. In particular, the front end does not process an instruction sequence. From the programmer's point of view, the front end subsystem is a large block of memory that must be initialized to some specified values. This simple model is complicated by the fact that the memory block is not contiguously populated, not all address locations are both read and write, and there are certain rules about the order and procedure in which these addresses are to be initialized. On another level, the programmer is faced with a variety of radar signal-processing algorithms that yield filter coefficients and beamforming weights. An understanding of how this data is combined and reorganized to conform to the front end module structure is also required. Although both of these areas have been touched upon in Section 2, this chapter consolidates that information and includes all of the additional information that is required for "programming" the front end.

The adaptive nulling processor, which resides in the same card cage as the front end modules, acts as the front end host processor (the VMEbus master) in the integrated signal processor test bed. Prior to this integration, a Sun workstation configured with a VMEbus repeater card was used as the host processor. Programs were written in the C programming language to configure the front end in a variety of test modes. This portion of the development effort has been completed, and the front end subsystem has since been integrated with the adaptive nulling processor. The final software developed to support the front end will execute on a Motorola DSP56001 processor. Even though this code is likely to be written in DSP56000 assembly language (the C compiler is very inefficient), referencing the original C programs may be useful.

3.1 Configuring the FIR Filters

This section is dedicated to the FIR filter portion of each front end module. The task of configuring the filters is, for the most part, limited to loading the eight A100 devices on each of the eight front end modules. However, unless the front end module Discrete Control Register (DCR) is loaded properly, the filters will not operate correctly. Instructions on programming the DCR are deferred to Section 3.2.

3.1.1 Front End Filter Addresses

Each A100 has 64 coefficient registers and two mode control registers, which must be loaded via the VMEbus interface. The front end subsystem is mapped into the 24-bit address space of the VMEbus and accepts 16-bit data transfers only. Figure 23, which is actually a subset of Figure 22, shows how the base address of each of the 64 A100s can be derived.

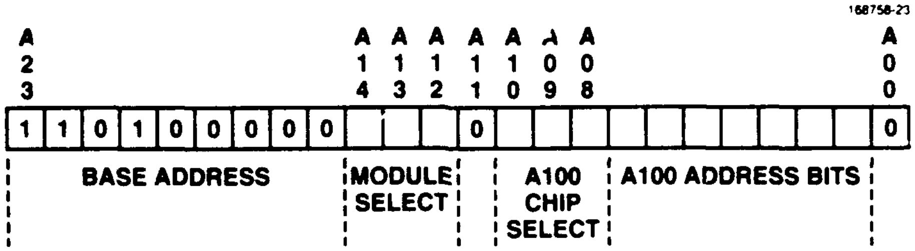


Figure 23. Front end filter address definition.

The LSB of the address word is used by the VMEbus to address data on a byte-by-byte basis.[5] Because all bus transactions with the front end modules are 16 bits wide. the LSB is always zero.

The base address that points to the start of the subsystem address space is D00000 Hex. The Module Select field identifies which module responds to the bus operation. as shown in Table 11.

TABLE 11

Decoding of Module Select Field in Address

Module Number	A14	A13	A12
1	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

[5] Actually, the address line A00 does not exist in the VMEbus Specification. and two data strobes are used to encode the value of A00. This arrangement is transparent to the programmer.

When addressing the A100s, address bit A11 is always zero. (When A11 is one, the beamformer dual-port RAMs or the DCR may be addressed.) The eight A100s on the front end module are selected by the A100 Chip Select Field according to Table 12.

TABLE 12

Decoding of A100 Chip Select Field in Address

A100 Number	A100 Designation	A14	A13	A12
1	I/Q1	0	0	0
2	I/Q2	0	0	1
3	I/Q3	0	1	0
4	EQPC1 (High)	0	1	1
5	EQPC2	1	0	0
6	EQPC3	1	0	1
7	EQPC4	1	1	0
8	EQPC5 (Low)	1	1	1

Address bits A07 through A01 are used by the A100 device to select the various coefficient and control registers. Table 13 presents the internal use of these seven address bits by the A100. The address in the table is given with respect to address bits A07 through A01 only and will have a zero appended as the LSB when used by the VMEbus. For complete definitions of these registers and their use, the reader is referred to the Inmos literature [4].

With the information provided thus far, it is now possible and desirable to set up a table of **DEFINE** statements in the software. This table should include the base address value **BASE** set equal to D00000, along with the A100 designations and register locations as described in the tables. A simple equation can then be used to compute all A100 addresses as needed:

```
address = (BASE + (module_id * 4096) + (a100_chip * 256) + (REG * 2))
```

where **module_id** will be equal to the integers from 0 to 7, **a100_chip** can either be integers from 0 to 7 or the A100 designations listed in Table 12 (which in turn are defined as equal to integer values), and **REG** is any of the registers specified in Table 13 (which are defined as the appropriate numeric addresses).

TABLE 13

A100 Memory Map

Register	Hex Address	Function
CCR(0-31)	20-3F	Current Coefficient Registers
UCR(0-31)	00-1F	Update Coefficient Registers
SCR	40	Static Control Register
	41	Unused location
ACR	42	Active Control Register
	43	Unused location
TCR	44	Test Control Register
DIR	48	Data Input Register
DOL	4A	Data Output Register (LSW)
DOH	4B	Data Output Register (MSW)

All VMEbus transactions must be 16 bits wide, must be written to the VMEbus 24-bit address space, and the VMEbus Address Modifier bits, AM5 to AM0, must equal 39 Hex (Standard Non-Privileged Data).

3.1.2 Front End Filter Data

I/Q Filter Coefficient Arrangement. The inphase and quadrature separation filter is designed as a 48-tap complex FIR filter [1]. The 48 complex filter coefficients, I(0) through I(47) and Q(0) through Q(47), are distributed among the three I/Q A100s in an interleaved manner. That is, A100 I/Q1 is loaded with the coefficients I(3n) and Q(3n) where n = 0,1,2,...,15. A100 I/Q2 is loaded with coefficients I(3n+1) and Q(3n+1) where n = 0,1,2,...,15. Finally, A100 I/Q3 is loaded with coefficients I(3n+2) and Q(3n+2) where n = 0,1,2,...,15. The I (inphase or real) part of the filter coefficients is loaded into the even-numbered Current Coefficient Registers (CCRs), i.e., CCR(0), CCR(2), CCR(4),...,CCR(30). The odd-numbered CCR registers, CCR(1), CCR(3),..., CCR(31) are loaded with zeros. Likewise, the Q (quadrature or imaginary) part of the filter coefficients is loaded into the even-numbered Update Coefficient Registers (UCRs), and the odd-numbered UCRs are loaded with zeros. Refer to Section 2.2.1 for an understanding of how this arrangement of coefficients implements the 48-tap filter. Figure 11 shows a partial demonstration of this technique. Table 14 gives a complete map of the 48-tap filter arrangement. The filter taps are indexed from 0 to 47, and the inphase and quadrature halves of each coefficient are designated with the notation I() and Q(), respectively.

TABLE 14

I/Q Filter Coefficient Arrangement

CCR/UCR Index	I/Q1 CCR	I/Q1 UCR	I/Q2 CCR	I/Q2 UCR	I/Q3 CCR	I/Q3 UCR
0	I(0)	Q(0)	I(1)	Q(1)	I(2)	Q(2)
1	0	0	0	0	0	0
2	I(3)	Q(3)	I(4)	Q(4)	I(5)	Q(5)
3	0	0	0	0	0	0
4	I(6)	Q(6)	I(7)	Q(7)	I(8)	Q(8)
5	0	0	0	0	0	0
6	I(9)	Q(9)	I(10)	Q(10)	I(11)	Q(11)
7	0	0	0	0	0	0
8	I(12)	Q(12)	I(13)	Q(13)	I(14)	Q(14)
9	0	0	0	0	0	0
10	I(15)	Q(15)	I(16)	Q(16)	I(17)	Q(17)
11	0	0	0	0	0	0
12	I(18)	Q(18)	I(19)	Q(19)	I(20)	Q(20)
13	0	0	0	0	0	0
14	I(21)	Q(21)	I(22)	Q(22)	I(23)	Q(23)
15	0	0	0	0	0	0
16	I(24)	Q(24)	I(25)	Q(25)	I(26)	Q(26)
17	0	0	0	0	0	0
18	I(27)	Q(27)	I(28)	Q(28)	I(29)	Q(29)
19	0	0	0	0	0	0
20	I(30)	Q(30)	I(31)	Q(31)	I(32)	Q(32)
21	0	0	0	0	0	0
22	I(33)	Q(33)	I(34)	Q(34)	I(35)	Q(35)
23	0	0	0	0	0	0
24	I(36)	Q(36)	I(37)	Q(37)	I(38)	Q(38)
25	0	0	0	0	0	0
26	I(39)	Q(39)	I(40)	Q(40)	I(41)	Q(41)
27	0	0	0	0	0	0
28	I(42)	Q(42)	I(43)	Q(43)	I(44)	Q(44)
29	0	0	0	0	0	0
30	I(45)	Q(45)	I(46)	Q(46)	I(47)	Q(47)
31	0	0	0	0	0	0

EQPC Filter Coefficient Arrangement. The next set of front end module filters, A100s EQPC1 through EQPC5, performs two distinct filtering tasks. One set of FIR filter coefficients is designed to perform channel equalization, i.e., to correct for any channel mismatch in the analog radar receiver. The other set of FIR coefficients is designed to perform pulse compression, i.e., to match filter the incoming waveform with the transmit pulse waveform. Pohlig [1] describes the design of these filters. The total tap length of these two filters must not exceed 80 complex taps.[6] Once the two sets of filter coefficients have been computed, the two functions are convolved, resulting in a combined equalization and pulse compression filter. If the two filters have a combined total of 80 taps, then the process of convolving will result in a 79-tap filter. When using the A100s to filter complex data with complex coefficients, the cascade of A100s loses one tap. Thus, the five A100 cascade implements a 79-tap complex filter.

The EQPC filter coefficients are arranged in the five EQPC A100s in a complicated and nonintuitive manner. First of all, the five filters are cascaded to create a 160-tap (80-complex-tap) filter. The sequence of coefficient registers on each A100 is concatenated. However, due to the transversal filter architecture, EQPC5 (A100 number 7) contains the lowest order coefficients, EQPC4 (A100 number 6) contains the next lowest set of coefficients, and so on. Further, the EQPC filter coefficients are arranged in the cascade of 160 CCRs and 160 UCRs in a unique manner. Table 15 gives a partial listing of this arrangement. The 79-tap complex filter is assigned indices from 0 to 78, and the real and imaginary halves of the coefficients are designated by I() and Q(), respectively. From this table, a pattern can be observed and used to interpolate the unlisted coefficient register contents. How this arrangement of filter coefficients implements a complex FIR filter, operating on complex data, is difficult to see at first. Section 2.2.2 discusses this approach in part, and the reader is also referred to the Inmos literature [5]. (The rest is left as a proof for the reader.)

A100 Control Registers. As seen in Table 13, each A100 has six read/write registers in addition to the 64 coefficient registers. The Data Input Register (DIR) and the two Data Output Registers (DOL and DOH) allow the A100 memory interface to pass data through the filter. Because the front end modules are designed to use the dedicated input and output ports rather than the memory interface, use of these registers will not be described.

The Static Control Register (SCR) must be initialized to the appropriate values before operation of the A100. Table 16 defines the significance of each bit in the SCR. All reserved bits should be written as zero and will return an unspecified value when read. The A100 "Master" mode is used when data is input through the DIR. This mode allows one of the A100s in a cascade to drive the GO signal to the remaining A100s. On the front end modules, the GO signal is driven by external logic, and the A100s are always in slave mode (bit 0 equals 0). Bit 1 selects between

[6] The pulse compression filter nominally requires 48 taps, and the remaining 32 taps are allotted for equalization.

TABLE 15

EQPC Filter Coefficient Arrangement

A100 Designation	Filter Tap	CCR/UCR Number	CCR Contents (Real)	UCR Contents (Imaginary)
EQPC5	0	0	0	-Q(0)
	1	1	I(0)	I(0)
	2	2	Q(0)	-Q(1)
	3	3	I(1)	I(1)
	4	4	Q(1)	-Q(2)
	5	5	I(2)	I(2)
	6	6	Q(2)	-Q(3)
	7	7	I(3)	I(3)
	8	8	Q(3)	-Q(4)
	9	9	I(4)	I(4)
	10	10	Q(4)	-Q(5)
	11	11	I(5)	I(5)
	12	12	Q(5)	-Q(6)
	13	13	I(6)	I(6)
	14	14	Q(6)	-Q(7)
	15	15	I(7)	I(7)
	16	16	Q(7)	-Q(8)
	17	17	I(8)	I(8)
	18	18	Q(8)	-Q(9)
	19	19	I(9)	I(9)
	20	20	Q(9)	-Q(10)
	21	21	I(10)	I(10)
	22	22	Q(10)	-Q(11)
	23	23	I(11)	I(11)
	24	24	Q(11)	-Q(12)
	25	25	I(12)	I(12)
	26	26	Q(12)	-Q(13)
	27	27	I(13)	I(13)
	28	28	Q(13)	-Q(14)
	29	29	I(14)	I(14)
	30	30	Q(14)	-Q(15)
	31	31	I(15)	I(15)
EQPC4	32	0	Q(15)	-Q(16)
	33	1	I(16)	I(16)
	34	2	Q(16)	-Q(17)
	35	3	I(17)	I(17)
	36	4	Q(17)	-Q(18)
	37	5	I(18)	I(18)
	38	6	Q(18)	-Q(19)
	39	7	I(19)	I(19)
	40	8	Q(19)	-Q(20)
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
EQPC1	150	21	Q(74)	-Q(75)
	151	22	I(75)	I(75)
	152	24	Q(75)	-Q(76)
	153	25	I(76)	I(76)
	154	26	Q(76)	-Q(77)
	155	27	I(77)	I(77)
	156	28	Q(77)	-Q(78)
	157	29	I(78)	I(78
	158	30	Q(78)	0
	159	31	0	0

TABLE 16

Format of the Static Control Register

Bit	Usage	Front End Module Value
(LSB) 0	Master not Slave	0 (Slave)
1	Input Data Source	0 (From Data Input Port)
2	Bank Swap Mode	1 (Continuous Swapping)
3	Reserved	0
4	Output Range Selection	0 (Select bits 7-30)
5		0
6	Reserved	0
7	Reserved	0
8	Coefficient Size	1 (16-bit Coefficients)
9		1
10	Fast Output Mode	0 (Normal Output Mode)
11	Reserved	0
12	Reserved	0
13	Reserved	0
14	Reserved	0
(MSB) 15	Reserved	0

the dedicated input port or the DIR as the input data source. The front end modules always use the dedicated input port (bit 1 equals 0). Swapping the data between the CCRs and UCRs can be performed every other data sample by setting bit 2 equal to 1. Because all A100s on the front end modules are used for complex filtering, this is the normal mode of operation.

The internal A100 multiply accumulators compute a 35-bit result. Because the A100 only outputs 24 bits, the user can select the 24-bit field according to Table 17. The latter two options imply sign extension of the result. The front end subsystem may use any of these options, and the choice may differ between the I/Q filters and the EQPC filters, but not between A100s within a filter structure. (Table 16 gives a default value of 00 for these control bits to select bits 7 to 30 of the data field.)

The A100s can process the 16-bit input data with 8-, 12-, or 16-bit coefficients. The front end subsystem algorithms [1] require 16-bit coefficients for all filtering operations. Therefore, bits 8 and 9 are always 1. The A100 supports a "Fast Output Mode," which outputs the high and low

TABLE 17

Specification of the Output Data Field

SCR(5-4)	Data Field
00	(7-30)
01	(11-34)
10	(15-38)
11	(20-43)

halves of the output word in rapid succession. This mode is not required or used by the front end subsystem.

The A1C0 Active Control Register (ACR) contains status and control bits that are likely to be accessed during operation of the device. The first bit (bit 0) is used to swap the CCR and UCR contents on a one-time basis. When this bit is set to one, the A100 will swap coefficient banks on the next data cycle and reclear the bit. This bit is not used in conjunction with the continuous swapping mode and thus is not used in the front end. The next two bits (bits 1 and 2) enable certain error conditions to cause the output Error signal to be activated. The front end system does not monitor the Error signal. It is a system requirement that all filters be designed to avoid overflow conditions: therefore, these bits are also not used. The remaining bits in the ACR are reserved. Therefore, the ACR should be initialized to all zeros upon power up and can then be ignored.

The third A100 control register is the Test Control Register (TCR). Bit 2 of this register can be used to override bits 4 and 5 of the SCR in selecting the output data range. If this bit is set to one, the output data field is -1 to 22. This range implies a zero LSB and can be useful in verifying correct operation of the entire dynamic range of the A100. The remainder of the TCR is reserved. The TCR should be initialized to all zeros after power up and then ignored. The A100 can and will power up in a proprietary test mode. Writing zeros to the TCR will return the A100 to normal operating mode. This step is therefore an absolute requirement. *This aspect of the A100 is not documented by Inmos.*

In summary, after power up, all 64 A100s in the front end subsystem should be initialized as follows. The ACRs and TCRs should be written with all zeros. All SCRs should be loaded with some predesignated value that is likely to be the same for all 64 devices and is nominally 0304 Hex. The 32 CCRs and 32 UCRs on each of the 64 A100s will be loaded with the filter coefficients. This data will vary from board to board and from chip to chip.

3.2 The Discrete Control Register

This section describes programming the front end module's Discrete Control Register (DCR). As described in Section 2.6.3, the DCR is a 16-bit register on each front end module that is memory mapped into the VMEbus address space. This register should not be confused with the control registers contained within the A100s. There is only one DCR per front end module, and it is used to configure several programmable aspects of the front end design. The DCR is a read-only register. It is therefore recommended that the system software always maintain a local copy of the contents of the DCR.

3.2.1 Discrete Control Register Addressing

Although the DCR is only one 16-bit register, the front end module loosely decodes the VMEbus address so that a number of different addresses will result in a write operation to the DCR. A read operation to the DCR is not defined. The VMEbus address format for the DCR is shown in Figure 24. The bit positions marked with an "X" can be either zero or one. As before, all VMEbus transactions are 16-bits wide and the Address Modifier Bits must be 39 Hex.

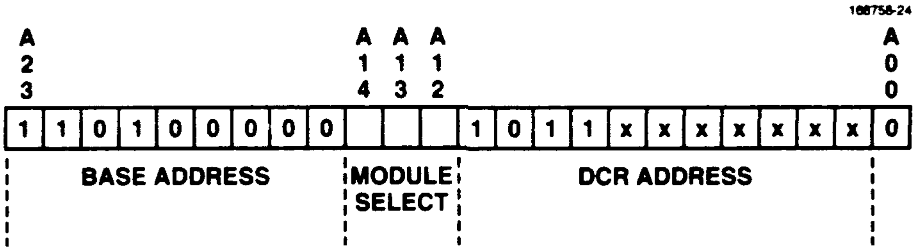


Figure 24. Discrete Control Register address definition.

If one value for the DCR address offset is selected (such as B00 Hex) and is defined in the program header, then the following equation can be used to compute each of the eight DCR addresses.

```
address = (BASE + (module_id * 4096) + DCR)
```

Recall that `module_id` points to the eight modules, according to Table 11, and that **BASE** is defined in Figure 24.

3.2.2 Discrete Control Register Data

The format of the DCR was first presented in Table 10.

Bits 0 through 11 are used in conjunction with the Data Input Zeroing function to allow only a prespecified number of nonzero data samples to be input in each PRI. This number will be programmed according to the experiment being run and will probably be the same for all eight modules. In the normal mode of operation, when data input zeroing is not used, these bits can be set to all ones. This configuration does not disable the function but will require 4096 input samples in a PRI before activating it.

The bit designated COEFSEL (bit 13) is used as an address bit into the beamformer dual-port RAM. This bit selects between the two alternate banks of beamformer coefficients and will be toggled frequently. In normal operation the beamformer coefficients will be updated every 10 ms. This updating requires the loading of the unselected 16 memory locations and then toggling this bit, on all eight modules, during a transmit pulse.

Bit 14 of the DCR, called TDMSELB/AN, selects which one of the two redundant TDM buses is to be used. In normal operation, both buses are functioning, and the state of this bit is irrelevant. In a deployed system, if diagnostics were to determine a chronic TDM bus failure, the alternate TDM bus could be selected.

The final bit in the DCR is CHDISABL (bit 15). If set to one, this bit disables operation of a front end module at the TDM bus transmit logic. One of the anticipated front end module failure modes is the failure of the module to properly recognize its TDM bus time slot. A module might attempt to broadcast data in several or all of the time slots. Because both TDM buses would be equally affected, this would result in a single-point failure of the entire processor; therefore, the ability to "turn off" the faulty module was incorporated. If a module is turned off and does not transmit data during its TDM bus time slot, the beamformers and adaptive nulling processor will generally observe data from the previous time slot. The presence of this data is due to the inherent capacitance of the bus. The adaptive nulling processor must compute the nulling weights to zero out this data.

3.3 Loading the Beamformer Weights

This section describes loading the beamformer dual-port RAM with the beamformer coefficients. First, the address field is defined and rules for writing to these addresses are described. Later, the distribution of the complex weights between the front end modules is explained.

3.3.1 Beamformer Weight Addresses

Each front end module contains either the inphase or quadrature half of a beamformer. Each half beamformer contains a 2-kword dual-port RAM, which is only partially mapped into the VMEbus address space. Within this mapping, two sets of 16 memory locations are defined for the alternate banks of beamformer coefficients. The selection between these banks is made according to the state of bit 13 in the DCR. This selection is transparent to the VMEbus. Therefore, both banks of coefficient registers are mapped on top of each other in the first 16 memory locations defined for the dual-port RAM. The VMEbus format for addressing the dual-port RAM is shown in Figure 25.

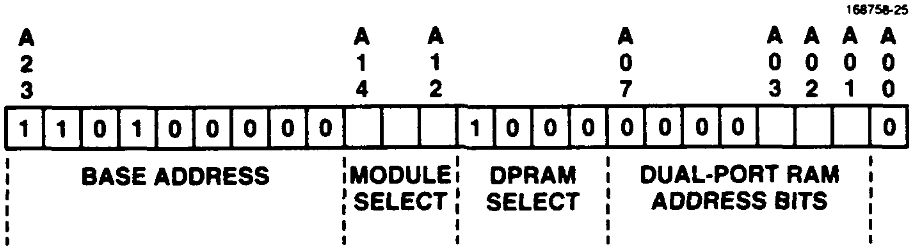


Figure 25. Beamformer dual-port RAM address definition.

Seven address bits (A07 to A01) are connected to address lines of the RAM. All of these locations can be written to and read from. However, only the first 16 locations (addressed by A03, A02, and A01) are defined for use. As always, only 16-bit operations are defined and the Address Modifier Bits must be 39 Hex.

Because of contention in the dual-port RAM, only the bank of locations *not* selected by bit 13 of the DCR can be written to. Thus, it will be necessary to keep track of the state of the DCR and to toggle this bit as appropriate when loading the beamformer coefficients. In normal operation of the system, the adaptive nulling processor will compute a new set of weights every 10 ms. These weights will be downloaded to the four beamformers in the coefficient bank that is not in use. Then, during the next transmit pulse (when data is not being processed), bit 13 is toggled in each of the eight DCRs.

If we define the variable **BF** to point to the start of the dual-port RAM memory space (800 Hex), then the following equation can be used to compute all of the beamformer coefficient register addresses.

```
address = (BASE + (module_id * 4096) + BF + (index * 2))
```

Recall that **BASE** is defined according to the figure above; **module_id** points to the eight modules according to Table 11 and to the beamformer halves, according to Table 4. The value of **index** will range from 0 to 15.

3.3.2 Distribution of the Beamformer Weights

The adaptive nulling processor samples data from the TDM bus and uses this data to compute nulling weights used by the beamformers [1]. Depending on the operating mode of the signal processor test bed, any combination of one to four beamformers may be used in a given experiment. Each beamformer requires a unique set of weights. Each set is composed of eight complex numbers with 16-bit real parts and 16-bit imaginary parts. If we define four sets of beamformer weights as A, B, C, and D, and we denote the real and imaginary parts of these weights as reA(), imA(), reB(), imB(), etc., then Table 18 demonstrates the arrangement of these weights in the eight front end modules. Section 2.4 explains how this arrangement of the beamformer weights results in the appropriate complex weighted sums.

TABLE 18

Arrangement of Beamformer Coefficients

Memory Index	Mod. 1 (BF1-I)	Mod. 2 (BF1-Q)	Mod. 3 (BF2-I)	Mod. 4 (BF2-Q)	Mod. 5 (BF3-I)	Mod. 6 (BF3-Q)	Mod. 7 (BF4-I)	Mod. 8 (BF4-Q)
0	reA(1)	imA(1)	reB(1)	imB(1)	reC(1)	imC(1)	reD(1)	imD(1)
1	reA(2)	imA(2)	reB(2)	imB(2)	reC(2)	imC(2)	reD(2)	imD(2)
2	reA(3)	imA(3)	reB(3)	imB(3)	reC(3)	imC(3)	reD(3)	imD(3)
3	reA(4)	imA(4)	reB(4)	imB(4)	reC(4)	imC(4)	reD(4)	imD(4)
4	reA(5)	imA(5)	reB(5)	imB(5)	reC(5)	imC(5)	reD(5)	imD(5)
5	reA(6)	imA(6)	reB(6)	imB(6)	reC(6)	imC(6)	reD(6)	imD(6)
6	reA(7)	imA(7)	reB(7)	imB(7)	reC(7)	imC(7)	reD(7)	imD(7)
7	reA(8)	imA(8)	reB(8)	imB(8)	reC(8)	imC(8)	reD(8)	imD(8)
8	-imA(1)	reA(1)	-imB(1)	reB(1)	-imC(1)	reC(1)	-imD(1)	reD(1)
9	-imA(2)	reA(2)	-imB(2)	reB(2)	-imC(2)	reC(2)	-imD(2)	reD(2)
10	-imA(3)	reA(3)	-imB(3)	reB(3)	-imC(3)	reC(3)	-imD(3)	reD(3)
11	-imA(4)	reA(4)	-imB(4)	reB(4)	-imC(4)	reC(4)	-imD(4)	reD(4)
12	-imA(5)	reA(5)	-imB(5)	reB(5)	-imC(5)	reC(5)	-imD(5)	reD(5)
13	-imA(6)	reA(6)	-imB(6)	reB(6)	-imC(6)	reC(6)	-imD(6)	reD(6)
14	-imA(7)	reA(7)	-imB(7)	reB(7)	-imC(7)	reC(7)	-imD(7)	reD(7)
15	-imA(8)	reA(8)	-imB(8)	reB(8)	-imC(8)	reC(8)	-imD(8)	reD(8)

4. MISCELLANEOUS SUBSYSTEM ISSUES

4.1 Hardware Description

As described in Sections 1 and 2, the front end subsystem has been partitioned into eight identical front end modules, which are constructed on 9U × 220-mm wirewrap boards (Mupac Corporation part number 203CD14C1M0M0). This board has three-level gold-plated wirewrap pins, an 8-HP front panel, and power pin commitments that are VMEbus compatible on the J1 connector only. The board is populated with 94 integrated circuits and several resistor packages. (The surface area is only about 60 percent full.) A photograph of the component side of one of the completed modules is shown in Figure 26.

Clearly, the size and weight of the system are dominated by the choice of these wirewrap boards. This selection is an artifact of the undefined nature of the design in the early stages of the program and the desire to maintain the same form-factor as other subsystem boards. The size and weight could be significantly reduced by migrating to a much smaller printed circuit board. Appendix B will discuss more aggressive next-generation packing schemes including Multi-Chip Modules (MCMs) and Application Specific Integrated Circuits (ASICs).

4.1.1 Backplanes

As with all 9U VMEbus compatible boards, there are three 96-pin DIN connectors on the back edge of the board. The top connector, denoted by P1, is completely dedicated to the VMEbus and conforms to the VMEbus standard in power commitments. The J1 backplane, which accepts all of the P1 connectors, is an off-the-shelf printed circuit board VMEbus backplane with the appropriate termination installed. The backplane has 20 slots, but due to the width of the wirewrap boards, only every other slot is used.

The middle connector, denoted by P2, does not conform to the VMEbus standard. The center column of pins is not connected to the upper 8 address bits and the upper 16 data bits of the A32/D32 VMEbus. The front end subsystem limits itself to the A24/D16 version of the VMEbus that is available on the P1 connector. However, because it was anticipated that a standard VMEbus card might be installed in the chassis to act as temporary bus master, the P2 VMEbus power commitment was preserved. The J2 backplane is a completely uncommitted wirewrap backplane. The appropriate power pins have been wirewrapped in place. The J2 backplane distributes the system clock, the PRI signal, the Datardy signal, and the TDM bus channel strobes. The Module ID bits are also wirewrapped for each front end slot on the J2 backplane. And finally, the J2 backplane is used to input data from the radar receiver (or TVG) and to output data along the Beamformer Output bus.

The third connector J3 is dedicated to the TDM bus. The primary TDM bus is on the A column of pins, the redundant TDM bus is on the C column of pins, and the B column of pins are all grounded. The J3 backplane is a printed circuit board that was constructed in-house. The A

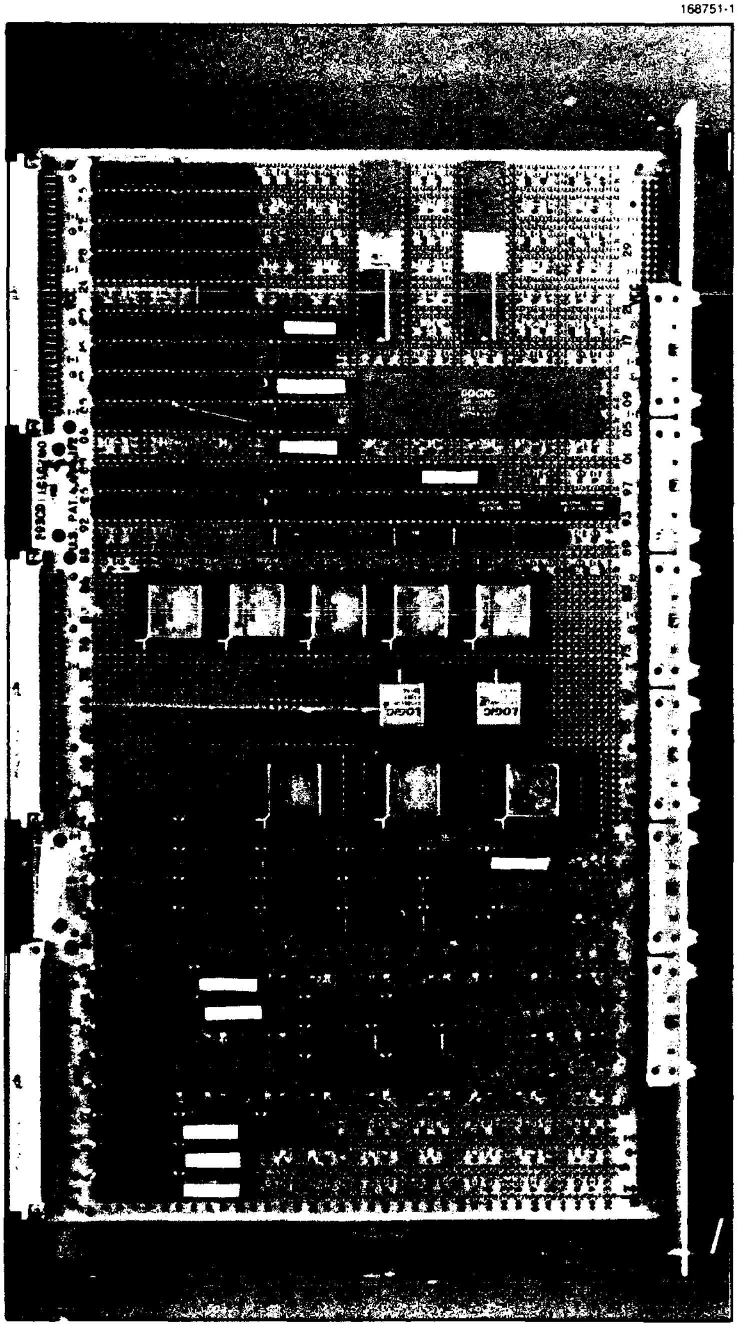


Figure 26. Photograph of a front end module.

and C rows of pins are bussed across to all of the connectors, and the B row is connected to the central ground plane. There is no termination on this backplane.

The specific pinout of the connectors can be seen on the schematics. The wiring of the J2 backplane can easily be deduced from these schematics.

4.1.2 Interconnect Harness

The front end modules receive their input data from the A column of pins on the P2 connector The A/D Interface section of this report describes how the interconnect between the two systems is accomplished via 34-conductor twisted-pair ribbon cables with 34-pin headers installed on both systems. The pinout for these connectors (and the cable) is given in Table 1. This section also describes the transmission of the PRI signal along a similar conductor with a single 16-pin header at each end. The signal processor test bed system rack has a panel mounted on it just below the front end card cage. This panel has eight 34-pin wirewrap headers (one for each A/D channel) and one 16-pin wirewrap header for the PRI signal. Special cables were fabricated to slip over the backside of these headers and to slip directly over the A column of pins on the wirewrap backplane. The eight Datardy signals, along with the PRI signal, are connected directly to the system clock board where they are buffered and then distributed to the rest of the system.

4.1.3 Test Connectors

Six 20-pin wirewrap headers have been installed on the front edge of the boards. These connectors are used for test purposes only and are mapped into the data collection pods of the Hewlett-Packard 16500A Logic Analysis System. The signals brought to these test connectors are mostly samples of the data stream as it proceeds through the module. Table 19 lists the six connectors, numbered from top to bottom, and describes their data contents and analysis method. Exact pinouts of these connectors can be seen on the schematics.

4.1.4 Front End Module Components

With the exception of the A100, selection of the components used in the front end subsystem design was based on the general availability, multiple vendors, radiation tolerance, and low power consumption. (The A100 was chosen based on how well its unique functionality matched the requirements of the front end subsystem.) Table 20 lists the front end module components along with the estimated power for each component and the anticipated radiation tolerance.

A typical, or even worst-case, power consumption estimate for the front end module has always been difficult to calculate. Inmos does not provide a typical power consumption estimate for the A100, and the worst-case estimate of 2.0 watts is significantly higher than our extrapolated measurements. The table values show a worst-case estimate of 16 watts for the eight A100s and a typical value of 9 watts for the remainder of the board. In reality, the front end modules have been measured at about 6.5 watts per board while data is continually being processed (5.2 watts in the

TABLE 19

Front End Module Test Connectors

Connector Designation	Data Available	HP16500A Analysis
TP1	Incoming A/D Data	State Mode
TP2	Data Stream after I/Q Filtering	State Mode
TP3	Data Stream after EQPC Filtering	State Mode
TP4	Data Stream off the TDM Bus	State Mode
TP5	Data Stream Out of the Beamformer Half	State Mode
TP6	Beamformer Output Bus & Misc. Timing Signals	State Mode and Timing Mode

quiescent state). To be conservative, the official power estimate is 8 watts per front end module. All power is drawn from a +5-volt supply.

A deployed version of the signal processor test bed may encounter any variety of hostile conditions. Many of these conditions (including humidity, shock and vibration, and temperature extremes) can be compensated for by the system packaging. The one environmental concern that is particularly difficult to shield against is total dose radiation. If the signal processor test bed was redesigned for an application requiring radiation resistance, the impact would be minimal. The only absolutely critical component is the Inmos A100. This device was tested at the Lincoln Laboratory Linear Accelerator and was shown to have a total dose resistance of at least 10 krad(Si). (These tests are documented in Appendix A.) As shown in Table 20, most of the components used in the design have a radiation resistance equal to or more than 10 krad(Si). While the radiation resistance of the Lattice GAL16V8 programmable logic devices (PLDs) is not known, these parts can easily be replaced with a variety of commercially available rad-hard PLDs. The same is true for the other components with unknown radiation resistance. Therefore, it can be concluded that the front end subsystem could be repackaged as a 10-krad(Si) total dose radiation resistant system with little or no impact to the design.

TABLE 20

Front End Module Component List

Q	Part Number	Package Size	Nominal Power W/IC	Total Dose Rad Tolerance
2	74F521	20 DIP	0.210	?
1	74LS240	20 DIP	0.125	1 MRAD
4	74LS244	20 DIP	0.500	1 MRAD
2	74LS374	20 DIP	0.250	1 MRAD
1	74LS38	14 DIP	0.005	1 MRAD
3	74ACT08	14 DIP	0.054	1 MRAD
1	74ACT138	16 DIP	0.042	1 MRAD
1	74ACT139	16 DIP	0.030	1 MRAD
4	74ACT161	16 DIP	0.132	1 MRAD
6	74ACT164	14 DIP	0.198	1 MRAD
6	74ACT273	20 DIP	0.216	1 MRAD
33	74ACT374	20 DIP	1.782	1 MRAD
2	74ACT377	20 DIP	0.120	1 MRAD
2	74ACT646	24 DIP	0.100	1 MRAD
2	74ACT74	14 DIP	0.054	1 MRAD
10	GAL16V8	20 DIP	4.050	?
2	IDT7132SA	48 DIP	0.650	≤6 KRAD[1]
8	IMSA100	84 PGA	≤2.0 ??	10 KRAD[2]
2	L4C381	68 PGA	0.250	?
1	MA7010DIP	64 DIP	0.200	100 KRAD[3]
1	ATT41MF	16 DIP	0.165	?

[1] 10 krad(Si) and 30 krad(Si) versions are claimed to be available.

[2] Tested at Lincoln Laboratory. See Appendix A.

[3] This version is no longer in production.

4.1.5 Programmable Logic Devices

The front end module design relies on 10 programmable logic devices (PLDs). Five of these devices implement the VMEbus address decoding and control. One controls the input data sequence, two provide control for the TDM bus, one controls the beamformer, and one sequences the Beamformer Output bus. All of these devices are Lattice GAL16V8s. They are used in simple modes of operation and are easily replaced with less complex fuse-type PLDs.

4.2 Time Delay Model

The front end subsystem and each front end module operate in a completely causal manner. The coefficients are preloaded, and the mode control registers are configured prior to operation of the system. After this point, the front end is not turned on or off or triggered for data processing in any way other than to have data pumped through it. For every three input data samples from the A/D interface (latched via the Datardy signal), one TDM bus output cycle will result. The delay between the trailing edge of the third Datardy pulse and the start of the TDM bus cycle is 42 24-MHz clock cycles.[7] The TDM bus cycle consists of eight channel strobes as shown in Figure 16 and lasts a total of 16 24-MHz clock cycles.

For every two TDM bus cycles, i.e., two sets of TDM bus channel strobes, one Beamformer Output bus cycle will result. This cycle appears as shown in Figure 21 and requires a total of 32 24-MHz clock cycles. The delay between the end of the second TDM bus cycle and the start of the Beamformer Output bus cycle is 9 24-MHz clock cycles. The net result is that for every six Datardy pulses, *the front end subsystem will generate one Beamformer* Output cycle or one complex output sample. (Each Datardy represents eight A/D samples, and each Beamformer Output cycle represents four beamformer samples.) The total delay from the sixth Datardy pulse to the beginning of the Beamformer Output bus cycle is 67 clock cycles.

The input-to-output delay introduced by the front end subsystem is, of course, longer than the delay between input samples. It is important to understand that each Beamformer Output sample corresponds to and is the result of six input A/D data samples. The beamformer sample that is being output at any given time is the result of input samples that arrived about 2.8 μs earlier. This information is important to both the adaptive nulling processor and the vector processors in synchronizing data samples with the PRI pulses. Figure 27 below attempts to characterize the time delay model of the front end subsystem.

[7] This TDM bus cycle transmits the inphase data with this delay. The quadrature sample that is broadcast during this cycle corresponds to the previous input sample. Thus, the quadrature sample delay is at least one 4.5-MHz clock cycle plus the 42 24-MHz clock cycles but has no upper bound.

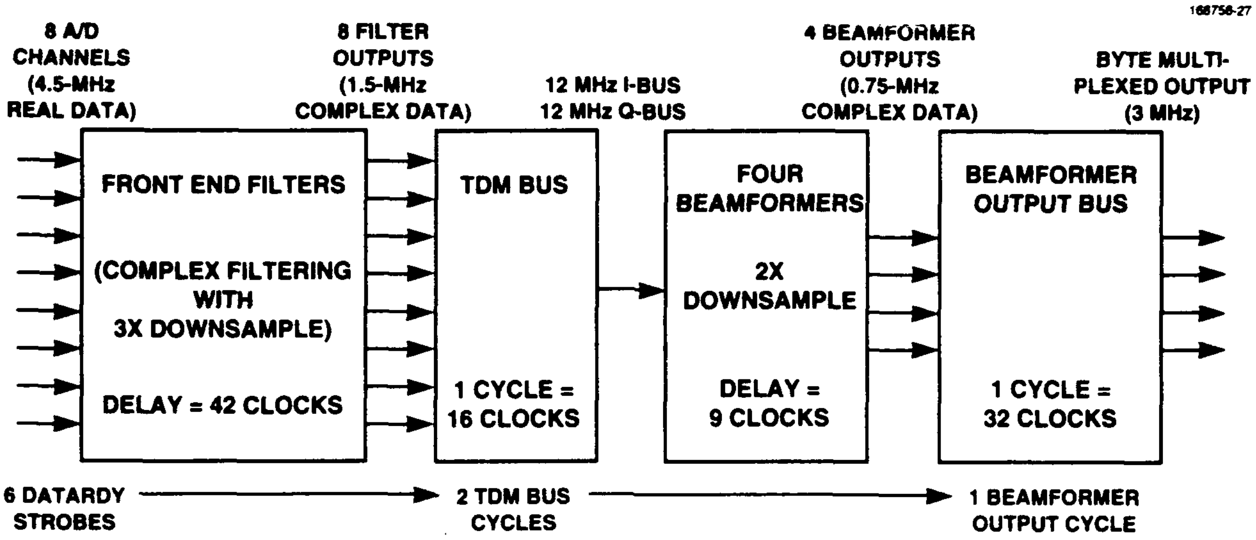

168756-27
8 A/D CHANNELS (4.5-MHz REAL DATA)
8 FILTER OUTPUTS (1.5-MHz COMPLEX DATA)
12 MHz I-BUS
12 MHz Q-BUS
4 BEAMFORMER OUTPUTS (0.75-MHz COMPLEX DATA)
BYTE MULTI-PLEXED OUTPUT (3 MHz)
FRONT END FILTERS
(COMPLEX FILTERING WITH 3X DOWNSAMPLE)
DELAY = 42 CLOCKS
TDM BUS
1 CYCLE = 16 CLOCKS
FOUR BEAMFORMERS
2X DOWNSAMPLE
DELAY = 9 CLOCKS
BEAMFORMER OUTPUT BUS
1 CYCLE = 32 CLOCKS
6 DATARDY STROBES
2 TDM BUS CYCLES
1 BEAMFORMER OUTPUT CYCLE

Figure 27. Time delay diagram of the front end subsystem.

4.3 System Clock Board

The most critical design requirement of the front end subsystem is that each of the eight front end modules must operate in lockstep to the 24-MHz system clock. Every stage in the design has to be carefully synchronized. Due to the throughput requirements of the system, there is no time to resynchronize the modules prior to the TDM bus. (They could be resynchronized prior to the Beamformer Output bus, but if the modules have maintained synchronization through the TDM bus and the beamforming process, then the rest is trivial.) Therefore, the only synchronization occurs at the data input. This requirement results in the addition of a system clock boaιd, which is centrally located in the front end subsystem card cage at slot number 6.

While the adaptive nulling processor and the vector processors require additional functionality from the clock board, the front end subsystem requires only three major functions from the clock board. These functions are the generation and distribution of the 24-MHz system clock, the synchronization and distribution of the Datardy signal, and the synchronization and distribution of the PRI signal. All three of these signals are driven down the backplane differentially via AT&T 41MP drivers, as shown for the system clock in Figure 15. They are received on each module with the 41MF receivers and terminated on the ends of the backplane.

The 24-MHz system clock must be at least 24 MHz. A smaller number of clock cycles can and will result in the random dropping of the 4.5-MHz data samples. (Likewise, the 4.5-MHz Datardy pulses can be no greater than 4.5 MHz.) The system clock must have a duty cycle of 50 percent with a tolerance of ±5 percent. (This tolerance has not been measured and may be larger in reality.)

Each A/D module in the radar receiver and each A/D channel in the TVG system generate a data strobe to accompany its outgoing data. These strobes are guaranteed to be tightly synchronized but for reasons of fault tolerance are not combined into one single data strobe. If it were sufficient that these eight strobes were synchronized to each other, then each front end module could receive the data strobes directly from its mated A/D channel. However, because the radar receiver and TVG system are not synchronized to the front end 24-MHz system clock, this approach cannot be taken. All eight data strobes must be synchronized to the *same* 24-MHz clock. The 4.5-MHz clock will flow in and out of phase with the 24-MHz clock, and data strobes occurring right on the edges of the 24-MHz clock will have uncertain results. Therefore, this synchronization of the data strobes with the 24-MHz clock must be done in a central location, the system clock board. In doing so, the clock board should use some fault-tolerant scheme of selecting which data strobe (from which channel) to use. The scheme should allow for failure of one or more channels without losing data strobes for all eight modules. (In the current incarnation of the system clock board, this requirement is deferred, and the trivial selection of the channel 1 strobe is made.) The resulting synchronized data strobe is called Datardy and driven down to both ends of the backplane.

When the front end subsystem is interfaced to the radar receiver, as opposed to the TVG system, some additional processing of the data strobes is required. These additional requirements are described in Section 2.1. Briefly, this processing consists of inverting the data strobes and stripping off the first two strobes occurring at the beginning of each PRI. This task is accomplished within a single PLD. When the front end is interfaced to the TVG system, an alternate PLD is installed.

The PRI signal is received by the front end subsystem along a separate twisted-pair cable. This signal is delivered to the clock board where it is buffered, synchronized to the 24-MHz clock, and driven down to both ends of the backplane. The synchronization of the PRI signal to the system clock is not really necessary but is done for historical reasons. (As shown in Figure 7, there is a minimum of 100 ns dead time on either side of the PRI pulse where no data strobes are occurring.)

An alternate approach to using a centrally located system clock board is to put the necessary clock and synchronization functions on all eight of the front end modules. The adaptive nulling processor could then select, via the VMEbus and the DCR, which module is to act as system master. This module would drive the critical signals down the backplane and the remaining front end modules would tristate their clock, Datardy, and PRI drivers and receive these signals as they do now. This approach has the advantage of improved fault tolerance but the disadvantage of a significant increase in hardware.

APPENDIX A
RADIATION TOLERANCE TESTING OF THE INMOS A100

This appendix documents the testing of the radiation resistance of the INMOS A100 digital filter I.C. using the MIT Lincoln Laboratory Radiation Test Facility.

Eight of the bulk silicon CMOS A100s were examined in the lab prior to radiation testing. Variations observed among the devices could be characterized by slight differences in their steady state supply current. The two devices with the highest and lowest supply current were chosen for testing. Both of these devices were standard commercial temperature range (0 to 70 deg Celsius) parts with lot code Δ8719 and with Lincoln Laboratory-assigned serial numbers 1 and 3.

Functional testing of the A100s throughout the test procedure was performed by the IMS B009 Evaluation Board purchased from Inmos. The board was configured in Mode C with switch SW2 in the up position. The evaluation board runs under an IBM PC/AT host, and Mode C allows for direct control of the A100 devices from the PC Bus. The setting of SW2 selects an operating clock speed of 20 MHz for the A100s. This is the maximum specified input clock speed.[8] The test conditions are closely matched to the expected A100 mode of operation in the signal processor. The IMS B009 Test Software, also provided by Inmos, performed a number of data transfer operations as well as a simple convolution test.

The A100 under test was extended up out of the PC chassis along a printed circuit extender card, Lincoln Laboratory part number 27PW14195. Six 0.01-microfarad bypass capacitors were added to the extender card between each of the six power and ground pins of the A100. In addition, the power pins of the UUT were connected and isolated in order to measure supply current during testing.

The A100 was situated in the direct line of the electron beam, 12 inches back from the opening. All testing was performed with 1.5 MeV electrons at a dose rate of about 1 krad per minute.[9] The Radiation Test Facility has been calibrated such that a fluence of 3×10^{13} e/cm^2 represents 1×10^6 rad(Si) or 10 kGy(Si). The units under test were subjected to and tested at total dose increments of 1, 3, 10, 15, and 20 krads(Si) equivalent dose. All functional tests and supply current measurements were taken while the beam was deflected away from the UUT. Annealing time between exposures was conducted with the device under bias and at room temperature.

For doses up to 3 krads(Si) there were no changes in static supply current and no functional failures. At 10 krad(Si) total dose exposure, one of the units failed the functional test in one out

[8] The front end subsystem actually uses a later generation part that can be clocked at speeds up to 30 MHz.

[9] This dose rate is much higher than typically encountered in a space environment.

of five trials, and both units showed an increase in static supply current of about 300 percent. Annealing trends were observed through a rapid decrease in static supply current after the initial measurements. The failed functional test was unrepeatable once annealing had begun. It is our position that the damage that anneals away while the part is under bias is a dose rate artifact and should not be considered in predicting the total dose performance of the part to ambient radiation.

The first repeatable functional failures were observed after a total dose exposure of 15 krads(Si). These failures were accompanied by a supply current increase of nearly five times the initial value. Annealing of the supply current was once again dramatic, but functional operation did not return immediately. The supply current had annealed to within 10 percent above its initial value and functionality returned 18 hours later.

An additional 5 krads(Si) of exposure resulted in the loss of functionality for the remainder of testing (another 18-hour annealing period). A similar supply current increase was observed but with a slower annealing curve.

Subsequent testing of the parts with no bias and 1-hour exposure to 70 deg Celsius resulted in complete recovery from functional failures and a return of the static supply current to near its pretest value. This observation would support a conclusion that with the 1.5-MeV electron radiation, the effects were predominantly ionizing and not bulk damage.

Figures A-1 and A-2 plot the supply current versus the total dose for each of the two units tested. The markers indicate the results of the functional tests at the end of each exposure period. Figures A-3 and A-4 plot the supply current as a function of time to demonstrate the annealing properties of each of the units tested. The markers indicate total dose exposure at the start of each annealing interval.

It can be concluded that the bulk silicon IMS A100 is at least a 10-krad(Si) total dose part. This radiation resistance qualifies it for a two-year mission in low-earth orbit, such as that encountered by a space-based radar.

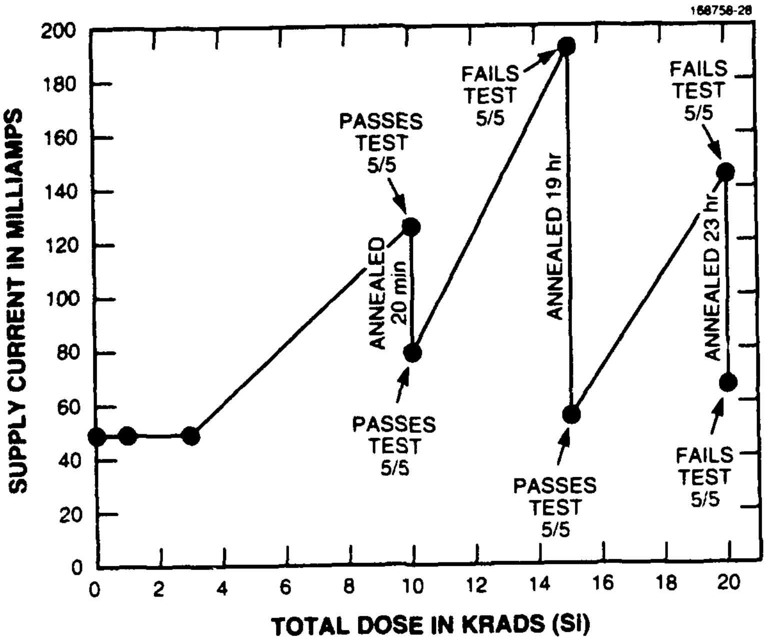


Figure A-1. Total dose effects on Inmos A100 sample number 1.

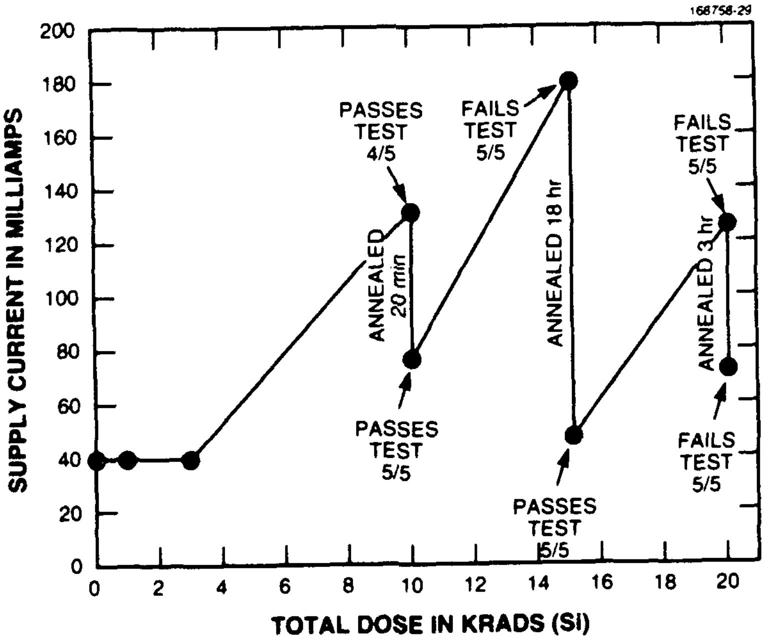


Figure A-2. Total dose effects on Inmos A100 sample number 2.

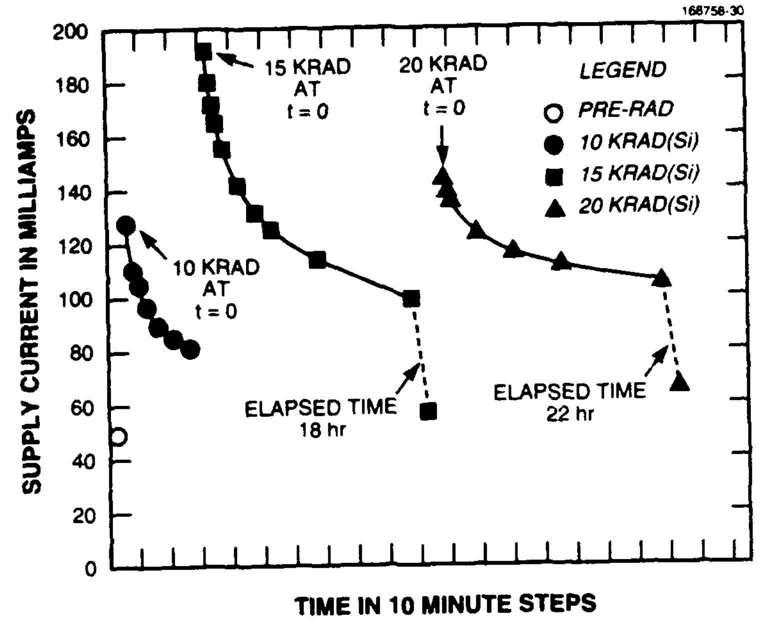


Figure A-3. Annealing trends on Inmos A100 sample number 1.

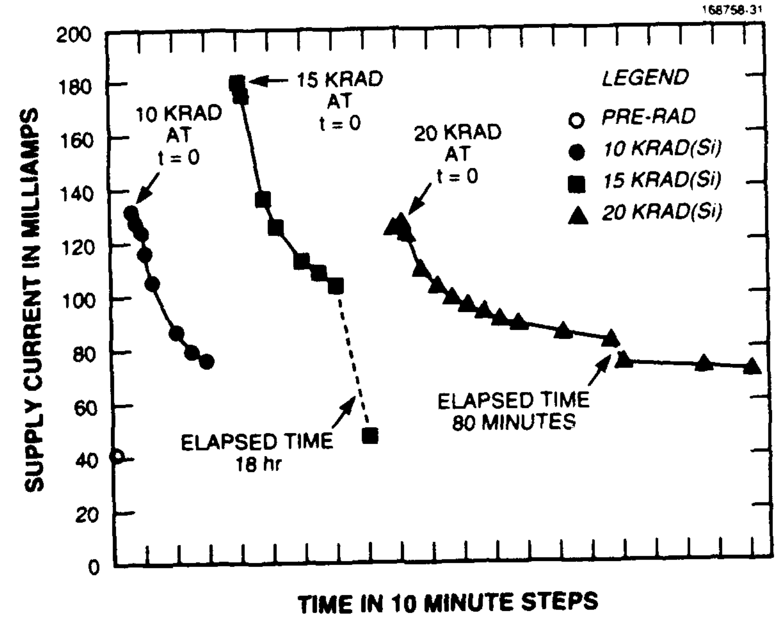


Figure A-4. Annealing trends on Inmos A100 sample number 2.

APPENDIX B
NEXT-GENERATION ISSUES

Two somewhat divergent paths can be pursued in developing a second-generation front end subsystem. One approach freezes the current architecture and follows up with compact packaging. This path will support a near-term system deployment and effective demonstration of the technology development. The other path expands the architecture to accommodate a larger number of channels (there has been interest expressed in a 64-channel system) and to improve the generality and flexibility of the design. This may require a deferment of the new packaging technologies and a delay in the demonstration of a deployable system, but the result may be a far more capable signal processor. Section B.1 discusses several options in next-generation packaging. These discussions will be in the context of the current architecture and design only. Section B.2 will discuss some of the issues of expanding the current architecture and make recommendations for a next-generation architecture.

B.1 Compact Packaging Schemes

B.1.1 Surface Mount Components

The current front end module is constructed on a wirewrap board with about 120 square inches of board space. These boards are about 60 percent populated. Without pushing the state of the art in electronics packaging in any way, the modules could be reconstructed on 80 square inch printed circuit boards. (The I/O could be reduced to two backplane connectors with data input on a front panel.) This packaging would considerably reduce the size and weight of the system. The technology is also readily available to replace the Dual In-line Packages (DIPs) with Surface Mount Components (SMCs). With this technology, we could reduce the board size even further and produce two-sided boards (one front end module on each side). This configuration would result in a four board front end subsystem. In moving from DIP technology to SMCs, we can expect a reduction factor of 8:1 in volume and 3:1 in weight [6]. (These ratios are taken with respect to the DIP printed circuit board implementation.) Thus, a truly compact 12-GOP system can be constructed with readily available packaging technologies.

B.1.2 Application Specific Integrated Circuits

Another strong candidate for improving the system packaging is replacing the glue logic with a few Application Specific Integrated Circuits (ASICs). This improvement would have the obvious advantage of considerably reducing the size of the system. Further, by reducing the number of

ICs, the reliability of the system is increased.[10] The reduced interconnect also increases reliability and reduces weight. And finally, the transition to ASICs reduces the system power requirements. A significant percentage of the front end power consumption is a result of the large number of output drivers on the MSI logic. These output drivers will not be necessary within an ASIC. An attempt to partition the front end module into five ASICs is shown in Table B-1. An overly conservative estimate of about 10,000 gates per ASIC was assumed. Some additional glue logic was also preserved. The result of this analysis was a total of 27 ICs. With state-of-the-art ASIC technology, this number could easily be cut in half. The entire design, excluding A100s, could probably fit in one 100,000 gate chip.

This transition to VLSI technology does introduce some risk because a partial redesign of the system is required. However, the advantages of reduced size, reduced weight, reduced power consumption, and increased reliability make it an irresistible option.

In Section B.2 the expansion of the current architecture to a 64-channel system is discussed. If this approach is pursued, the economic trade-off of one-time ASIC design costs versus production costs will favor the ASIC implementation.

B.1.3 Multi-Chip Modules

A recent advance in electronics packaging is the Multi-Chip Module (MCM). This approach involves mounting a number of silicon die, sans package, directly onto a silicon substrate. The result is an incredibly small and lightweight system with electronic packaging at the IC level eliminated. The reduced number of interconnections also increases reliability. An MCM system implementation will weigh about 6 percent of its DIP counterpart [6]. Considering that the front end might also have its IC count reduced via ASIC designs, this percentage will be even lower.

Next-generation packaging of the front end subsystem could begin with two parallel efforts. An MCM could be developed to include the eight A100s and the appropriate glue logic, while the remainder of the module could be redesigned in an ASIC format. The results of these two efforts could then be integrated on very small printed circuit boards. Eventually, the entire design could be incorporated on a single MCM. Figure B-1 shows the reduction of the front end module as these packaging concepts are applied.

B.2 Subsystem Redesign Issues

A redesign of the front end subsystem would consider two major architectural enhancements. The first enhancement would expand the system to accommodate 64 A/D channels. The second enhancement would improve the generality of the system to allow for changes in the A/D sampling

[10] A simple model for electronic systems reliability is the sum of the failure rates for each IC. Thus, system reliability can be increased by improved fabrication processes and reduced IC count.

TABLE B-1

ASIC Front End Module Component List

Q	Part	Contents	I/O Pins
1	ASIC 1	(12) FACT273	70
	Input Logic	(2) FACT08	
		(1) FACT74	
		(1) LS240	
		(1) GAL16V8	
		(1) FACT 164	
1	ASIC 2	(2) L4C381	65
	Adders	(3) FACT374	
2	ASIC 3	(12) FACT374	70
	TDM Buses	(1) FACT08	
	A and B	(1) FACT74	
		(1) GAL16V8	
		(1) FACT139	
1	ASIC 4	(1) MAC1010	65
	Beamformer	(2) DPRAMS	
		(1) GAL16V8	
		(1) FACT161	
		(1) FACT164	
1	ASIC 5	(5) GAL16V8	80
	VME and Misc.	(2) FACT377	
		(3) FACT161	
		(2) F521	
8	A100		
3	LS244		
6	LS374		
1	GAL16V8		
2	FACT646		
1	LS38		

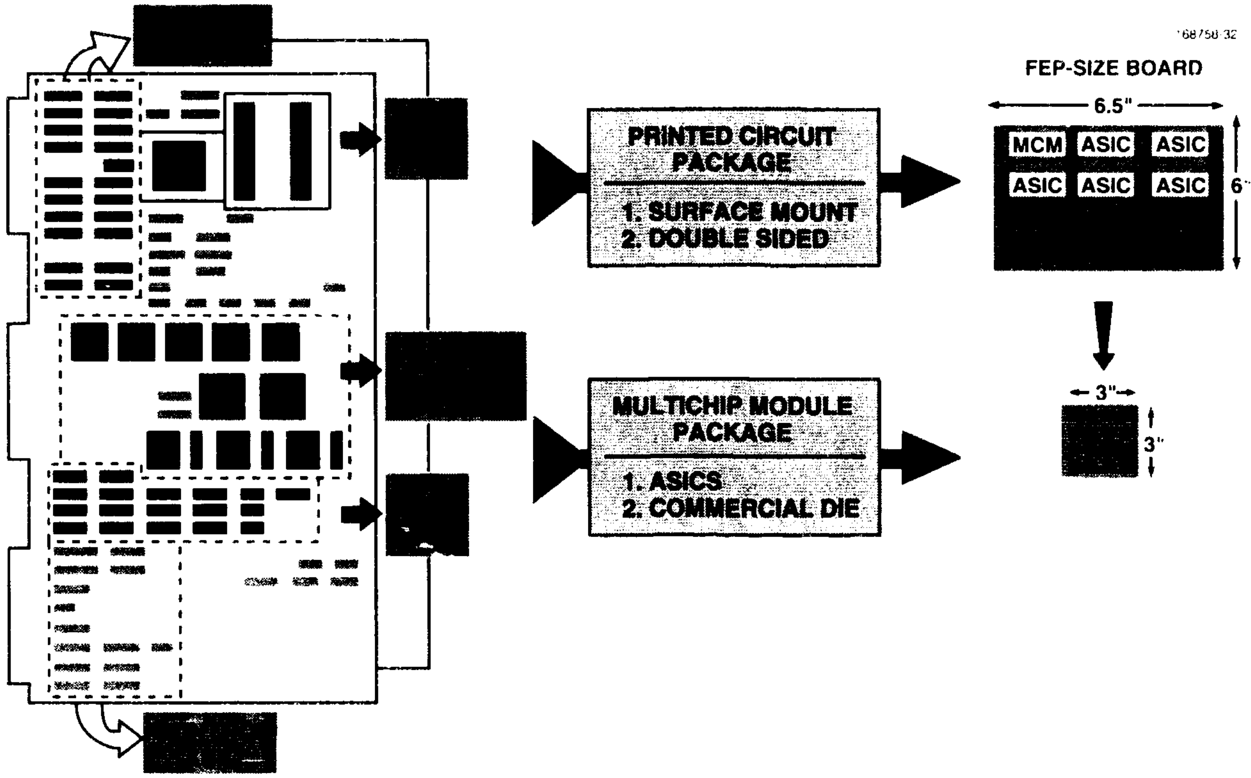


Figure B-1. Future packaging of front end module.

rates, the bandwidth of the radar, the bit precision, and the transmit waveform. While it may never be possible to accommodate all of these generalities in one reasonably sized system, it is still useful to keep them in mind while redesigning the system.

Recent developments in related programs at Lincoln Laboratory indicate that future generations of the adaptive nulling processor will be able to compute nulling weights, in real time, for a system with 64 degrees of freedom. The next-generation signal processor should therefore be able to apply those 64 nulling weights to 64 channels of radar data. This expansion requires an eightfold increase in the number of front end filter modules. However, referring to the system block diagram of Figure 1, an expansion of the system would also require an eightfold increase in bandwidth on the FDM bus. This expansion is clearly not achievable. One approach is to transmit the output data from each set of FIR filters in a bit serial format. This yields 256 I/O pins (assuming inphase, quadrature, clock, and ground for each channel) with data transmission at 24 MHz. Although this data rate and interconnect scheme are achievable, the method would require a massive redesign of the beamformers.

An expansion from 8 to 64 front end filters does not mean that the number of beamformers would also be increased. The best approach is to decouple the beamformer design from the front end filters. Each beamformer, constructed on its own board, would input 256 lines from the filter modules. This architecture would require 256 serial-to-parallel converters, probably developed with large ASICs. The massive computational requirements would necessitate several stages of multipliers in a pipelined format. This method is not technically difficult but would require significant real estate. A clever ASIC design, which could be parallelized and pipelined, might solve this problem.

The parallel A/D input interface, after expansion to 64 channels, would require so much cabling that the wires might outweigh the electronics. Bit serial data transmission between the A/D modules and the front end filters would also be necessary.

The pulse compression filtering can be performed before or after the nulling weights have been applied. In the current front end subsystem, combining the pulse compression filters with the equalization filters proved convenient. It was simpler to cascade additional A100s prior to beamforming than to configure a separate bank of A100s after the beamformers and before the Beamformer Output bus. The trade-off is that compression filtering must be performed on all eight A/D channels instead of the four beamformer channels. In a 64-channel system, with four beamformers, the trade-offs clearly lean in the opposite direction. The pulse compression filters should be moved after the beamformers.

With the removal of the pulse compression filters, a redesign of the entire filter architecture should be considered. The current design is specifically tailored to a 4.5-MHz A/D sample rate with a 1.5-MHz intermediate frequency and a 1-MHz bandwidth. These radar parameters are prone to change. The A100 can be run up to 30 MHz with faster versions on the horizon. Other silicon houses are producing alternate FIR filter chip designs. If the A100 feature of complex filtering via bank swapping is not available, then separate filter paths for inphase and quadrature can be designed. A user-programmable downsample circuit could follow the combined I/Q and Equalization filters. This improvement in generality will be worth any extra components it adds.

Consider the block diagram of the next-generation system shown in Figure B-2. Although this does not represent a comprehensive view of system requirements, it demonstrates the additional problem of transmitting the huge volume of filtered data to the adaptive nulling processor as well as the beamformers. To maintain generality, the adaptive nulling weights should be applied to the same data that was used to calculate the weights. This approach will require some type of buffer memory to delay the data stream prior to the beamformer. Perhaps if this memory had a second access port, the adaptive nulling processor can acquire its data samples directly from the data stream.

This section has presented a variety of issues that may be considered in redesigning the front end subsystem. This list is neither complete nor absolute. Many of the issues will be refined when a set of system requirements and performance goals are outlined. Where applicable, alternate design approaches have been identified and are merely suggestions and do not have the benefit of careful analysis or simulation.

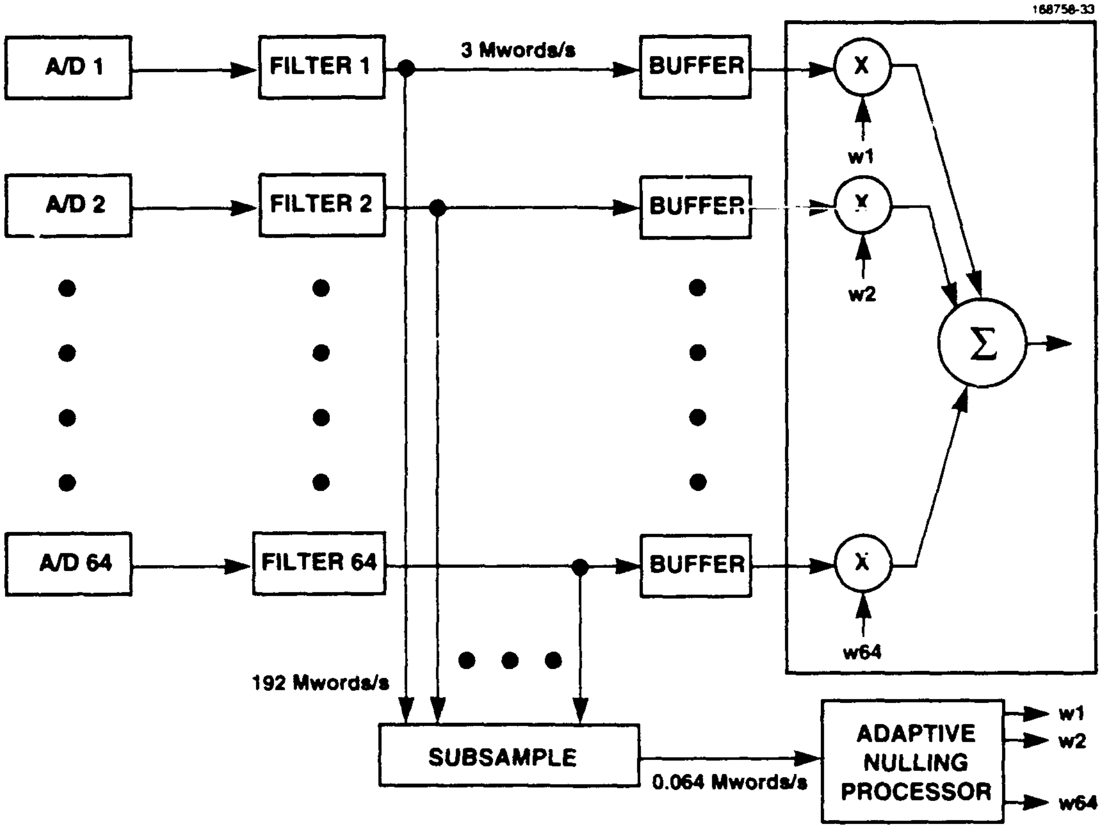


Figure B-2. Issues for next-generation front end design.

GLOSSARY

ACR	Active Control Register (on Inmos A100)
A/D	Analog to Digital (conversion)
ABEL	A PLD Programming Language and Design Tool
ALU	Arithmetic Logic Unit
ASIC	Application Specific Integrated Circuit
CCR	Current Coefficient Register (on Inmos A100)
CMOS	Complementary Metal-Oxide Semiconductor, an IC technology
CPI	Coherent Processing Interval (of radar signal processor)
DCR	Discrete Control Register (on front end module)
DIP	Dual In-line Package (for integrated circuit)
DIR	Data Input Register (on Inmos A100)
DOH	Data Output High register (on Inmos A100)
DOL	Data Output Low register (on Inmos A100)
EQPC	Equalization and Pulse Compression filters
FIR	Finite Impulse Response (digital filter)
GOP	Giga-OPerations (one billion arithmetic operations per second)
I	Inphase (or real) component of a complex signal
I/Q	Inphase/Quadrature filters
IC	Integrated Circuit
LSB	Least Significant Bit
MAC	Multiply Accumulator
MCM	Multi-Chip Module
MHz	Megahertz (one million cycles per second)
MSB	Most Significant Bit
MSI	Medium Scale Integration
PGA	Pin Grid Array, IC package

GLOSSARY
(Continued)

PLD	Programmable Logic Device
PRI	Pulse Repetition Interval (of the radar)
Q	Quadrature (or imaginary) component of a complex signal
RAM	Random Access Memory
SBC	Single Board Computer
SCR	Static Control Register (on Inmos A100)
SMC	Surface Mount Component
TCR	Test Control Register (on Inmos A100)
TDM	Time-Division-Multiplexed bus
TTL	Transistor-Transistor Logic. an IC technology
TVG	Test Vector Generator
UCR	Update Coefficient Register (on Inmos A100)
UUT	Unit Under Test
VLSI	Very Large Scale Integration
VMEbus	Industry standard bus interface (Versa Module Europe bus)

REFERENCES

1. S.C. Pohlig. "Digital signal processing for space-based radar," MIT Lincoln Laboratory, Lexington, Mass., Project Report SRT-30, (8 September 1988).

2. The VMEbus Specification, Revision C.1, Motorola, Inc. (October 1985).

3. C.B. Robins, "A Test Vector Generator for a radar signal processor," MIT Lincoln Laboratory, Lexington, Mass., Technical Report 827, (7 February 1991).

4. Inmos Ltd., "IMS A100 Cascadable Signal Processor," Inmos Limited, Bristol, England (February 1988).

5. H. Yassaie, "Complex (I & Q) processing with the IMS A100," IMS A100 Application Note 4, Inmos Limited, Bristol, England (Not dated).

6. D.E. Weidler (private communication, 1990).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
12 July 1991

3. REPORT TYPE AND DATES COVERED
Technical Report

4. TITLE AND SUBTITLE
A Front End Filter Subsystem for an
Adaptive Radar Signal Processor

5. FUNDING NUMBERS
C — F19628-90-C-0002

6. AUTHOR(S)
C.B. Robins

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Lincoln Laboratory, MIT
P.O. Box 73
Lexington, MA 02173-9108

8. PERFORMING ORGANIZATION REPORT NUMBER
TR-912

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
HQ AF Systems Command AFSC/XTKT
Andrews AFB
Washington, DC 20334-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ESD-TR-90-187

11. SUPPLEMENTARY NOTES
None

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This report documents the front end subsystem portion of an eight channel, adaptive nulling, radar signal processor test bed. The subsystem implements in excess of 12 billion operations per second on incoming data to effect signal conditioning through time-domain filtering. The hardware has been prototyped on eight circuit boards, each about 120 in^2, which are roughly half-populated. Compact packaging schemes are discussed in one of the appendices. This effort represents a demonstration of the technology required for a variety of on-board signal processors. Wherever possible, fault-tolerant design techniques and radiation-tolerant components have been used.

The front end subsystem receives eight channels of sampled data from the eight radar receiver A/D modules at the conversion rate of 4.5 MHz. The front end employs finite impulse response (FIR) filters to perform inphase and quadrature signal separation, channel equalization, and pulse compression. The coefficients for these filters are programmable via a VMEbus compatible interface. The front end also includes four digital beamformers; each computes a weighted sum of the eight channels of data on a sample-by-sample basis. The weights, also downloaded via the VMEbus interface, are computed and applied to perform adaptive nulling. The four resulting data streams, each with a rate of 0.75 million complex samples per second, are then output to the vector processor portion of the test bed for Doppler processing.

14. SUBJECT TERMS
radar signal processing
finite impulse response (FIR) filters
inphase/quadrature filter
channel equalization
pulse compression
digital beamformer
adaptive nulling

15. NUMBER OF PAGES
96

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
None